Graduado en Ingeniería Informática

Universidad Politécnica de Madrid

Facultad de Informática

TRABAJO FIN DE GRADO

Desarrollo de un proceso de normalización semántica de bases de datos en ensayos clínicos

Autor: Guetón Padrón Sánchez

Tutor: David Pérez del Rey

POLITÉCNICA
"Ingeniamos el futuro"

CAMPUS
DE EXCELENCIA
INTERNACIONAL

MADRID, ENERO 2015

Índice de contenidos

1 RESUMEN DEL TRABAJO REALIZADO

Este Trabajo de Fin de Grado ha sido realizado por el alumno de Grado en Ingeniería Informática de la Universidad Politécnica de Madrid, Guetón Padrón Sánchez en el primer semestre del curso 2014-2015. El tutor del proyecto ha sido el profesor David Pérez del Rey.

El trabajo se enmarca dentro de los proyecto INTEGRATE y EURECA, cuyo objetivo es el desarrollo de una capa de interoperabilidad semántica que permita la integración de datos e investigación clínica, proporcionando una plataforma común que pueda ser integrada en diferentes instituciones clínicas y que facilite el intercambio de información entre las mismas. De esta manera se promueve la mejora de la práctica clínica a través de la cooperación entre instituciones de investigación con objetivos comunes. En los proyectos se hace uso de estándares y vocabularios clínicos ya existentes, como pueden ser HL7 o SNOMED, adaptándolos a las necesidades particulares de los datos con los que se trabaja en INTEGRATE y EURECA. Los datos clínicos se representan de manera que cada concepto utilizado sea único, evitando ambigüedades y apoyando la idea de plataforma común.

El alumno ha formado parte de un equipo de trabajo perteneciente al Grupo de Informática de la UPM, que a su vez trabaja como uno de los socios de los proyectos europeos nombrados anteriormente.

La herramienta desarrollada, tiene como objetivo realizar tareas de homogenización de la información almacenada en las bases de datos de los proyectos haciendo uso de los mecanismos de normalización proporcionados por el vocabulario médico SNOMED-CT. Las bases de datos normalizadas serán las utilizadas para llevar a cabo consultas por medio de servicios proporcionados en la capa de interoperabilidad, ya que contendrán información más precisa y completa que las bases de datos sin normalizar.

El trabajo ha sido realizado entre el día 12 de Septiembre del año 2014, donde comienza la etapa de formación y recopilación de información, y el día 5 de Enero del año 2015, en el cuál se termina la redacción de la memoria.

El ciclo de vida utilizado ha sido el de desarrollo en cascada, en el que las tareas no comienzan hasta que la etapa inmediatamente anterior haya sido finalizada y validada. Sin embargo, no todas las tareas han seguido este modelo, ya que la realización de la memoria del trabajo se ha llevado a cabo de manera paralela con el resto de tareas. El número total de horas dedicadas al Trabajo de Fin de Grado es 324.

Las tareas realizadas y el tiempo de dedicación de cada una de ellas se detallan a continuación:

- Formación. Etapa de recopilación de información necesaria para implementar la herramienta y estudio de la misma [30 horas].

- Especificación de requisitos. Se documentan los diferentes requisitos que ha de cumplir la herramienta [20 horas].
- Diseño. En esta etapa se toman las decisiones de diseño de la herramienta [35 horas].
- Implementación. Desarrollo del código de la herramienta [80 horas].
- Pruebas. Etapa de validación de la herramienta, tanto de manera independiente como integrada en los proyectos INTEGRATE y EURECA [70 horas].
- Depuración. Corrección de errores e introducción de mejoras de la herramienta [45 horas].
- Realización de la memoria. Redacción de la memoria final del trabajo [44 horas].

This Bachelor Project has been performed by Guetón Padrón Sánchez, student of a Degree in Computer Science in the Informatics University (ETSIINF) of the Universidad Politécnica de Madrid (UPM). The project has been developed during the first semester of the 2014-2015 academic year. Professor David Pérez Del Rey has been the project's tutor.

This project belongs to the semantic interoperability layer developed in the European projects INTEGRATE and EURECA, which aims to provide a platform to promote interchange of medical information from clinical trials to clinical institutions. Thus, research institutions may cooperate to enhance clinical practice. Different health standards and clinical terminologies has been used in both INTEGRATE and EURECA projects, e.g. HL7 or SNOMED-CT. These tools have been adapted to the projects data requirements. Clinical data are represented by unique concepts, avoiding ambiguity problems.

The student has been working in the Biomedical Informatics Group from UPM, partner of the INTEGRATE and EURECA projects.

The tool developed aims to perform homogenization tasks over information stored in databases of the project, through normalized representation provided by the SNOMED-CT terminology. The data query is executed against the normalized version of the databases, since the information retrieved will be more informative than non-normalized databases.

The project has been performed from September 12th of 2014, when initiation stage began, to January 5th of 2015, when the final report was finished.

The waterfall model for software development was followed during the working process. Therefore, a phase may not start before the previous one finishes and has been validated, except from the final report redaction, which has been carried out in parallel with the others phases.

The tasks that have been developed and time for each one are detailed as follows:

- Training. Gathering the necessary information to develop the tool [30 hours].
- Software requirement specification. Requirements the tool must accomplish [20 hours].
- Design. Decisions on the design of the tool [35 hours].
- Implementation. Tool development [80 hours].
- Testing. Tool evaluation within the framework of the INTEGRATE and EURECA projects [70 hours].
- Debugging. Improve efficiency and correct errors [45 hours].
- Documenting. Final report elaboration [44 hours].

2 INTRODUCCIÓN

2.1 Planteamiento del problema

El presente Trabajo de Fin de Grado está enmarcado dentro de los proyectos de investigación de ámbito europeo *INTEGRATE* y *EURECA*. El objetivo de ambos proyectos es el desarrollo de una capa de interoperabilidad semántica, con el propósito de lograr la integración de datos clínicos y de investigación de ensayos clínicos. De esta manera, se proporciona una herramienta para el intercambio de dicha información, fomentando la colaboración entre clínicas investigadoras a gran escala.

Dicha capa de interoperabilidad semántica contiene entre otros elementos y servicios, un modelo común de datos. Se trata de una base de datos relacional basada en el modelo RIM del estándar HL7 v3 (Health Level Seven), en la que se almacenan los datos provenientes de ensayos e investigaciones clínicas llevadas a cabo en diferentes clínicas. Los datos almacenados incluyen información de los pacientes que han participado en dichas pruebas, así como los resultados obtenidos de observaciones y procedimientos realizados a cada uno de los pacientes.

El modelo común de datos es actualizado siempre que se requiera almacenar información cuya representación aún no ha sido modelada. Las actualizaciones, al igual que el modelo base, se llevan a cabo en base al modelo RIM mencionado anteriormente.

Los conceptos médicos almacenados en el modelo común de datos están codificados según una serie de terminologías médicas, en la que predomina SNOMED-CT, una de las terminologías más completas y extendidas en el campo de la informática biomédica. Pero otras terminologías como LOINC y HGNC también están presentes, completando el espectro del vocabulario necesario para representar los conceptos médicos con los que se trabaja en los proyectos en los que se enmarca este trabajo.

Uno de los objetivos más importantes de estos proyectos europeos, es la consulta precisa de los datos, para ello, SNOMED-CT proporciona mecanismos de homogenización de los conceptos, así, los conceptos pueden ser representados en una forma normal, aplicando una serie de reglas de transformación.

El objetivo principal de este Trabajo de Fin de Grado, es el de desarrollar una herramienta que se encargue de homogenizar los datos cargados en el modelo de datos común haciendo uso de los mecanismos de normalización mencionados anteriormente, para luego almacenarlos en otra base de datos implementada según el mismo modelo de datos común. De esta forma se obtendrá una base de datos en la que los conceptos médicos se encuentran almacenados según su forma normal. Esta base de datos normalizada será la utilizada para proporcionar servicios de acceso a los datos, que permitirán una recuperación más precisa gracias a la homogenización llevada a cabo por la herramienta normalizadora a implementar.

La capa de interoperabilidad semántica ofrece un servicio en el que se proporciona un método encargado de obtener la forma normal corta de un concepto dado, el cuál será usado por la herramienta a implementar en este trabajo. Según el resultado de dicha normalización, el proceso normalizador de bases de datos almacenará en las diferentes tablas del modelo común de datos, los conceptos que han resultado de generar su forma normal según corresponda, manteniendo la relación entre dichos conceptos y el paciente.

A su vez, el proceso de normalización, se encarga de una serie de tareas de homogenización de los datos más allá de la propia representación en forma normal, así por ejemplo, comprueba el valor de ciertos atributos de las tablas del modelo de datos común, de manera que si carecen de valor les asigna el correcto basándose en el tipo de datos y los valores que el modelo RIM de HL7 proporciona para dichos campos.

2.2 Objetivos

Como se ha mencionado en la sección anterior, el objetivo principal del Trabajo de Fin de Grado, es el desarrollo de un proceso de normalización semántica que se encargue de representar conceptos médicos de manera que se permita una consulta precisa y homogénea de dichos datos.

Para lograr el resultado esperado se han fijado una serie de objetivos, que serán necesarios alcanzar para cumplir con el propósito del trabajo. A continuación se muestra una lista con dichos objetivos y una descripción de los mismos:

- **Estudio del modelo de datos HL7 RIM.**
  El primer objetivo que ha de ser cumplido, es el del estudio del modelo común de datos. Para ello, en primer lugar, se ha de estudiar el modelo RIM de HL7 v3, un estándar de interoperabilidad para el intercambio electrónico de información clínica. Es importante conocer su estructura y el tipo de información que almacena, para luego entender el modelo común de datos utilizado en los proyectos *INTEGRATE* y *EURECA*, ya que está basado en el modelo RIM.
- **Estudio del mecanismo de normalización SNOMED Short Normal Form.**
  Antes de poder diseñar el proceso de normalización, es importante conocer los mecanismos de normalización de conceptos que ofrece el vocabulario biomédico

SNOMED-CT. Existen dos tipos de representación de forma normal, una larga y otra corta. En la plataforma en la que se incluirá este Trabajo de Fin de Grado, se utilizará la forma normal corta, la cual es obtenida a partir de la forma normal larga, eliminando una serie de relaciones redundantes.
La normalización puede dar lugar a diferentes situaciones, en las que se incluye, que el concepto a normalizar se mantenga inalterado, que sea modificado, y que aparezcan nuevos conceptos relacionados.

- **Estudio del enlace entre vocabularios biomédicos y el modelo HL7 RIM.**
  Como se ha mencionado en la subsección anterior, en el modelo común de datos se almacenan una serie de conceptos biomédicos codificados en diferentes vocabularios, predominando SNOMED-CT. Es necesario estudiar los métodos existentes para conocer qué decisiones se toman para almacenar dichos conceptos en las diferentes tablas de la base de datos. En la plataforma en desarrollo existen mecanismos encargados de esta tarea que han de ser estudiados.
- **Diseño e implementación de un proceso de normalización semántica basada en HL7 RIM y SNOMED Normal Form.**
  Una vez realizados los estudios previos, se procederá al diseño e implementación del proceso de normalización semántica. Haciendo uso de los servicios ya implementados en la plataforma y diferentes librerías Java, la herramienta puede ser desarrollada.
- **Evaluar el proceso de normalización con datos reales.**
  En la fase final o de pruebas, se evalúa el correcto funcionamiento del proceso de normalización haciendo uso de datos reales provenientes de clínicas de investigación biomédica. El objetivo final es obtener una base de datos en la que la información esté representada de manera normalizada para ofrecer una consulta homogenizada de los datos.

2.3 Organización de la memoria final

A continuación se muestra la estructura del presente documento, con una breve descripción de los apartados principales:

- **Resumen del trabajo realizado.** Se describe brevemente el trabajo realizado a lo largo del desarrollo de este Trabajo de Fin de Grado.
- **Introducción.** Se describe de manera general el Trabajo de Fin Grado, detallando los objetivos a alcanzar.
- **Estado de la cuestión.** Se presentan los estándares y vocabularios médicos utilizados, así como los servicios utilizados de los proyectos en los que se enmarca el trabajo.
- **Tecnologías empleadas.** Se detallan las diferentes herramientas y estándares utilizados para el desarrollo de la herramienta.

- **Especificación de requisitos.** Se incluye la descripción de los requisitos de la herramienta.
- **Diseño de la herramienta.** Las decisiones de diseño tomadas y la descripción del proceso de implementación se detallan en esta sección.
- **Implementación de la herramienta.** Recoge los detalles de desarrollo de la herramienta.
- **Resultados y pruebas.** Se presentan los resultados obtenidos y pruebas realizadas sobre el proceso de normalización. Dichas pruebas serán llevadas a cabo con datos reales.
- **Conclusiones y líneas futuras.** Sección final en la que se mostrará la consecución de los objetivos marcados, así como las posibles mejoras que se podrían llevar a cabo.

3 ESTADO DE LA CUESTIÓN

El desarrollo de la herramienta se enmarca dentro de los proyectos europeos *INTEGRATE* [1] y *EURECA* [2], cuyo objetivo principal es el desarrollo de una capa de interoperabilidad semántica para permitir un el intercambio de información clínica entre entidades investigadoras, fomentando la colaboración entre las mismas.

El mayor problema al que se enfrentan las clínicas colaboradoras a la hora de intercambiar información entre ellas es la heterogeneidad de los datos. La plataforma en desarrollo proporciona una visión estandarizada de la forma de almacenar y representar los datos, consiguiendo interoperabilidad entre los sistemas informáticos de los diferentes centros investigadores.

En las siguientes subsecciones, se presentará la manera en la que se integra el proceso de normalización en los proyectos europeos mencionados anteriormente, así como los vocabularios médicos en los que se codifican los diferentes conceptos médicos presentes en los datos generados y el modelo de datos común utilizado en la capa de interoperabilidad semántica.

3.1 Enfoque de integración dentro de los proyectos INTEGRATE y EURECA

Como ya se ha mencionado al comienzo de la sección 3 del presente documento, el proceso de normalización ha de integrarse dentro de la capa de interoperabilidad semántica desarrollado en los proyectos INTEGRATE y EURECA [3].

Entre los servicios desarrollados en el marco de estos proyectos se encuentra el Core Dataset Service, que sirve de contenedor de las terminologías médicas que se mencionarán en la sección 3.2.

Por otro lado, la capa de interoperabilidad semántica también proporciona herramientas de acceso a los datos mediante los servicios Data Access y Query Normalization. Este

último se encarga de, dado un concepto codificado en alguna de las terminologías médicas contenidas en el Core Dataset, proporcionar una consulta en lenguaje SPARQL que será ejecutada por el servicio Data Access, recuperando la información de pacientes que tengan algún tipo de relación con dicho concepto.

El último de los servicios presentes en los proyectos que se enmarca este Trabajo de Fin de Grado, es el Data Push. Su misión es almacenar datos provenientes de diferentes fuentes o recursos en el modelo común de datos. Dentro del servicio Data Push, se incluye un proceso de ETL que realiza las tareas de mapeo entre la información recibida y el modelo común de datos.

La figura 3.1 muestra la arquitectura de la capa de interoperabilidad semántica donde se ha de integrar el proceso de normalización semántica.

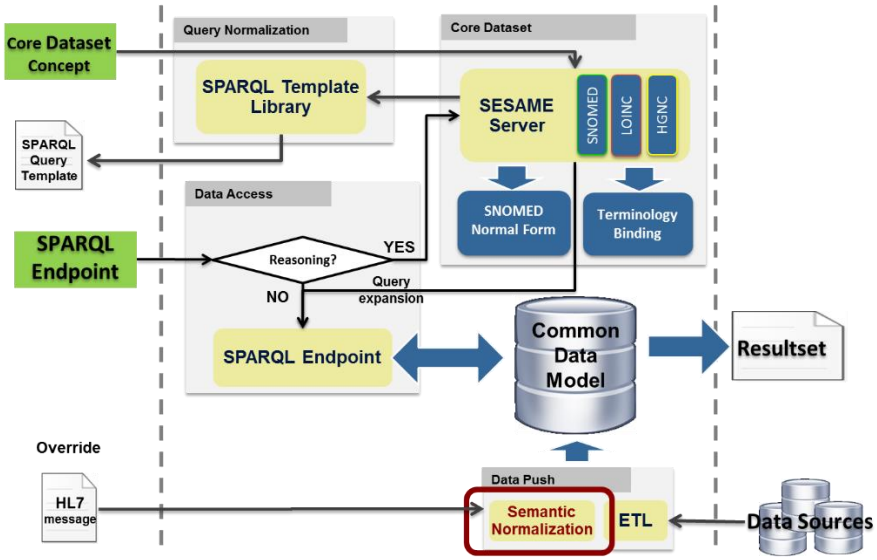


Figura 3.1: Capa de interoperabilidad semántica, proyectos INTEGRATE y EURECA

Como se puede observar en la figura anterior, la herramienta de normalización se integra dentro del servicio de almacenamiento de datos Data Push. De esta manera, la información presente en el modelo común de datos será almacenada tras ser sometida al proceso de normalización llevada a cabo por la herramienta desarrollada.

Una vez normalizados los datos, los servicios de acceso podrán trabajar con consultas en SPARQL más completas y precisas, ya que tras el proceso de normalización, se dispondrá de una mayor cantidad de información a recuperar.

3.2 Vocabularios médicos

3.2.1 SNOMED-CT

SNOMED-CT [4] es una de las terminología médicas más completas y extendidas que existen. Permite codificar datos clínicos proporcionando a los profesionales de la salud una manera de representar la información clínica de forma precisa e inequívoca.

Este vocabulario fue desarrollado originalmente por el *College of American Pathologists* (CAP). A partir de Abril de 2007, SNOMED pertenece al *International Health Terminology Standards Development Organisation* (IHTSDO), encargado de su mantenimiento y distribución.

SNOMED-CT proporciona mecanismos con el objetivo de facilitar la recuperación completa y de expresiones codificadas en este vocabulario. En la siguiente subsección se tratarán estos mecanismos.

3.2.1.1 Forma normal SNOMED

La forma normal de SNOMED [5] es una representación de conceptos biomédicos que se puede generar aplicando una serie de reglas lógicas sobre una expresión válida. Los conceptos pueden ser primitivos, o estar completamente definidos, en una expresión representada en forma normal, todos los conceptos presentes son primitivos, es decir, sus roles o atributos no expresan completamente su significado.

El proceso de normalización es recursivo, de modo que cualquier elemento de la definición de un concepto que se refiere a otro completamente definido, también es reemplazado por su forma normal.

En otras palabras, representar un concepto en su forma normal, implica reemplazar dicho concepto por otros primitivos que le definan, siempre y cuando el concepto original no sea ya primitivo. A su vez, los conceptos que definen al concepto original han de ser primitivos, en caso de que no lo sean, se les aplicará la forma normal.

Existen dos tipos de representación, la forma normal larga y la forma normal corta. La forma normal larga se obtiene de la misma forma que la larga, eliminando una serie de relaciones redundantes de la expresión final.

En la figura 3.2, se muestra un ejemplo de normalización de un concepto completamente definido, *fracture of femur*, codificado con la secuencia 71620000. Como se puede observar, la forma normal larga y la corta son iguales, ya que no hay relaciones redundantes.

Concept	71620000 | fracture of femur |
Definition *(distributed)*	116680003 | is a | = 64572001 | disease | {116676008 | associated morphology | = 72704001 | fracture | ,363698007 | finding site | = 71341001 | bone structure of femur | }
Long NF *(candidate)*	64572001 | disease | : {116676008 | associated morphology | = 72704001 | fracture | ,363698007 | finding site | = 71341001 | bone structure of femur | }
Short NF *(predicate)*	64572001 | disease | : {116676008 | associated morphology | = 72704001 | fracture | ,363698007 | finding site | = 71341001 | bone structure of femur | }

Figura 3.2: Forma normal del concepto fracture of femur

Como se puede observar, el concepto original se descompone en un concepto central o *focus concept*, surgido de la relación *is a*, que en este caso es *disease*, y en otros dos conceptos surgidos de las otras relaciones que definen el concepto, *associated morphology* y *finding site*, que son *fracture* y *bone structure of femur* respectivamente.

En la figura 3.3 se muestra un ejemplo el que la forma normal corta es distinta que la larga, ya que se eliminan algunas de las relaciones que definen el concepto original, en este caso *allergic asthma*.

Concept	389145006 | allergic asthma |
Definition *(distributed)*	116680003 | is a | = 195967001 | asthma | ,116680003 | is a | = 418168000 | disorder due to allergic reaction | ,42752001 | due to | = 419076005 | allergic reaction | {116676008 | associated morphology | = 26036001 | obstruction | ,363698007 | finding site | = 955009 | bronchial structure | }
Long NF *(candidate)*	195967001 | asthma | : 42752001 | due to | = 419076005 | allergic reaction | {116676008 | associated morphology | = 26036001 | obstruction | ,363698007 | finding site | = 955009 | bronchial structure | }
Short NF *(predicate)*	195967001 | asthma | : 42752001 | due to | = 419076005 | allergic reaction |

Figura 3.3: Forma normal del concepto allergic asthma

3.2.2 LOINC

Logical Observation Identifiers Names and Codes (LOINC) [6], se trata de un estándar para representar observaciones recogidas en laboratorios médicos. LOINC es una de las terminologías presentes en el Core Dataset Service. A diferencia de SNOMED, no presenta mecanismos de composición de términos, por lo que no será relevante estudiarlo en profundidad para desarrollar esta herramienta. Los conceptos

representados bajo esta terminología quedarán en variables de la base de datos normalizada.

3.2.3 HGNC

La tercera terminología médica incluida en el Core Dataset Service es *HUGO Gene Nomenclature Committee* (HGNC). Ésta proporciona un código o nombre único para todos los genes humanos conocidos, por lo que es una terminología sometida a actualizaciones siempre que sea descubierto un nuevo gen. Al igual que LOINC, HGNC no presenta un mecanismo de normalización, por lo que el proceso normalizador mantendrá estos conceptos invariables en la base de datos final.

3.3 Modelo RIM de HL7 v3

Health Level Seven (HL7) [7][8], es una organización sin ánimo de lucro, dedicada al desarrollo de estándares que faciliten el intercambio de información clínica de manera electrónica. Uno de los estándares más importantes publicados por esta organización ha sido *HL7 version 3 Normative Edition*, se trata de un estándar de mensajería basado en el *Reference Information Model (RIM)* [9]. El RIM, es uno de los componentes principales de HL7 v3, raíz de todos los modelos de información y estructuras desarrolladas como parte del estándar v3. Este modelo ha sido aprobado como estándar por el ANSI.

El RIM está formado por tres clases principales, éstas son:

- **Acto (*Act*).** Registro de algo que se está haciendo, se ha hecho, se puede hacer, está previsto hacer, o se ha solicitado hacer.
  A su vez, los actos pueden ser observaciones (*observation*), un acto que puede tener valores, o procedimientos (*procedures*), acciones realizadas sobre un sujeto, entre otros. Las administraciones de sustancia (*substanceAdministration*), son un tipo de procedimiento que consiste en la introducción o aplicación de un material a un sujeto.
- **Entidad (*Entity*).** Objeto o ente físico, grupo de objetos o entes físicos, o una organización capaz de participar en los actos mediante un rol específico.
- **Rol (*Role*).** Papel que una entidad desempeña en un acto.

Estas clases se relacionan entre sí a través de las siguientes clases:

- **Relaciones entre actos (*ActRelationship*).** Asociación directa entre dos actos.
- **Participación (*Participation*).** Asociación entre un acto y un rol. La entidad que desempeña el rol es el actor.
- **Enlace entre roles (*RoleLink*).** Asociación entre dos roles.

La siguiente figura muestra las clases y relaciones anteriores:

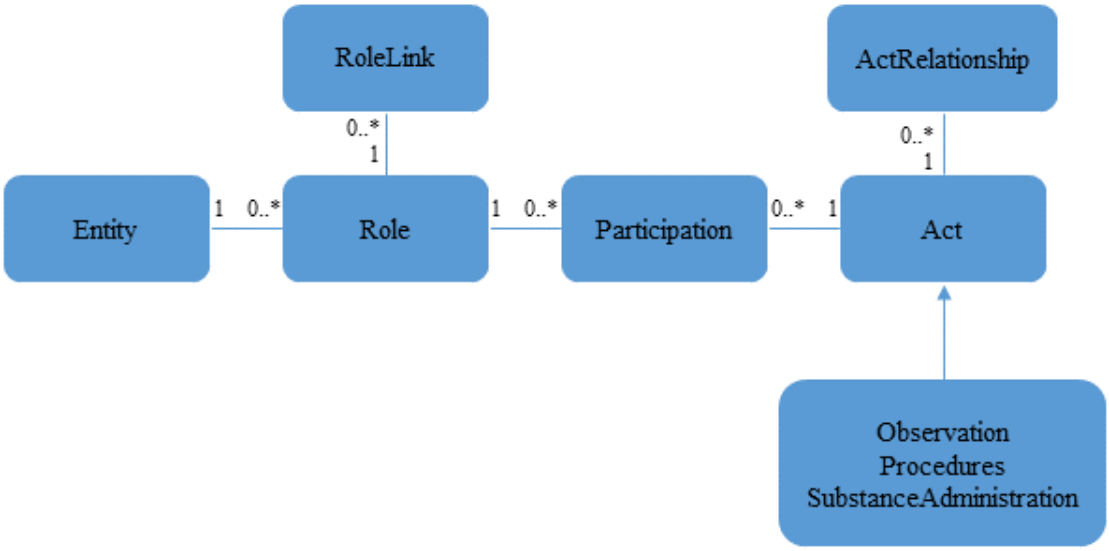


Figura 3.4: Clases y relaciones principales del modelo RIM de HL7

Debido a la amplitud del modelo RIM, en los proyectos en los que se enmarca este Trabajo de Fin de Grado, se tomó la decisión de adaptar el modelo a las necesidades específicas de dichos proyectos. En la figura 3.4, se muestra el esquema completo del modelo RIM de HL7.

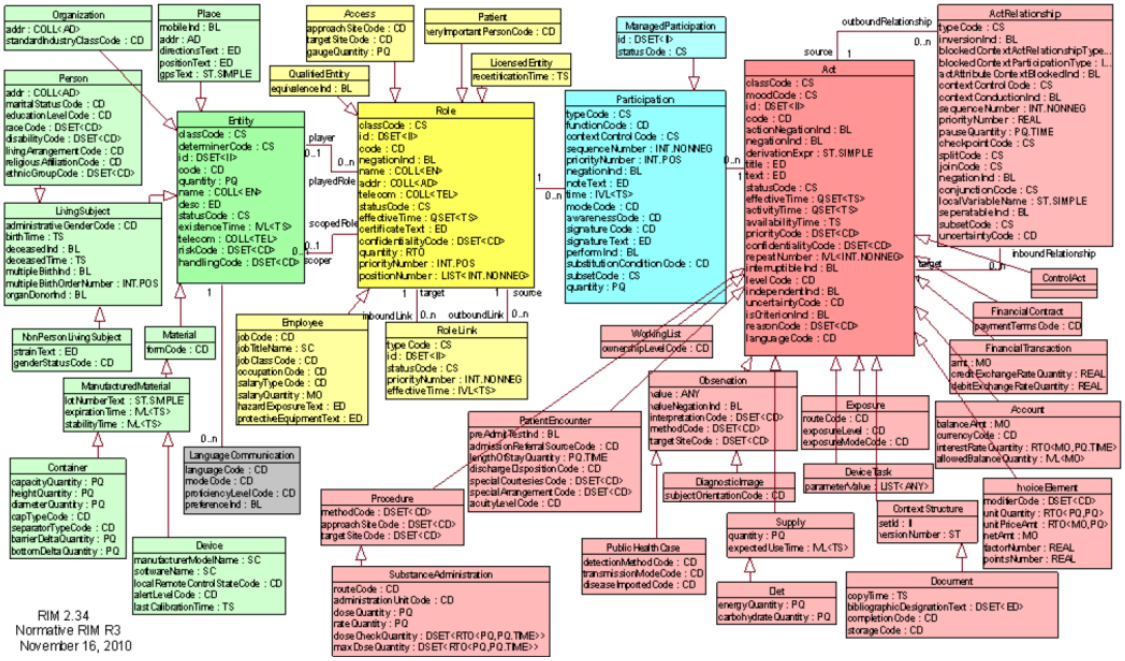


Figura 3.4: Esquema del modelo RIM de HL7

La figura 3.5 muestra el modelo de datos común utilizado en los proyectos en su versión 2.6.3. Como se puede observar, es un modelo reducido del RIM de HL7, ya que ha sido diseñado para que satisfaga las necesidades específicas de modelización de los datos

provenientes de las clínicas colaboradoras. Este modelo es actualizado siempre que surjan nuevas necesidades que impliquen la creación de nuevas tablas, relaciones o atributos de tablas.

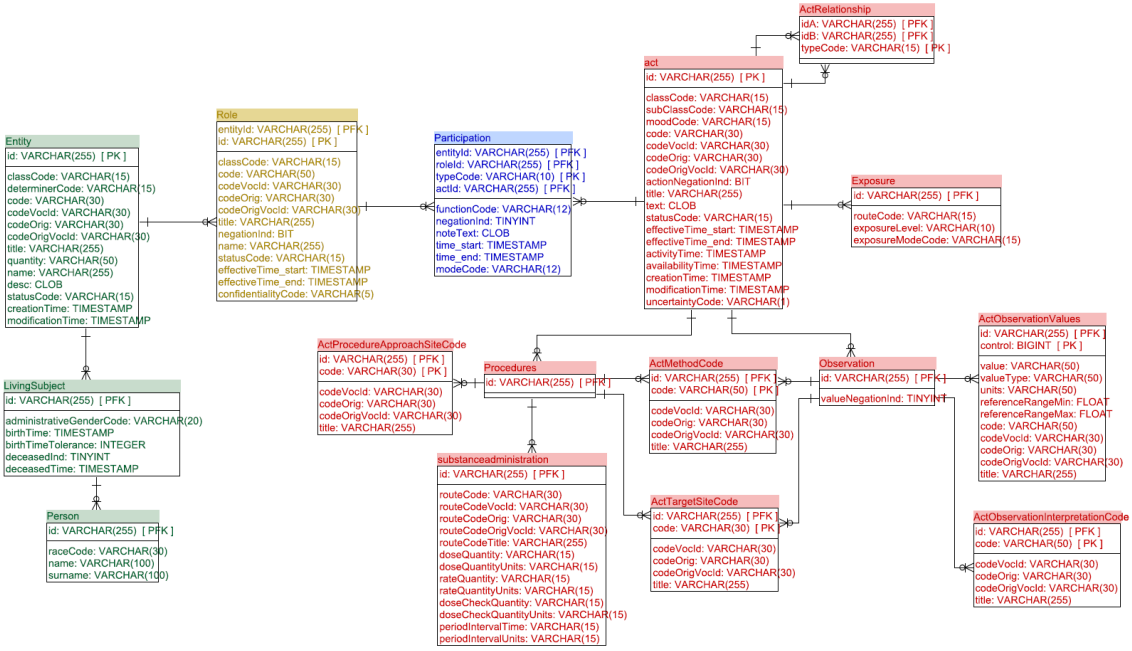


Figura 3.5: Common Data Model versión 2.6.3

3.4 Terminology binding

Una de las herramientas proporcionadas por el servicio Core Dataset, es el Terminology Binding. El objetivo principal de esta herramienta es enlazar las terminologías médicas utilizadas en el proyecto con el componente del modelo RIM de HL7.

En el caso de SNOMED-CT, el funcionamiento general del Terminology Binding es, realizar un análisis de la posición de los conceptos dentro de la jerarquía de términos presentes en este vocabulario para decidir a qué componente del RIM se corresponde (*Acto*, *Observación*, *Entity*,…).

Este proceso es básico para el desarrollo del proceso normalizador, ya que una vez obtenida la forma normal de un concepto en particular, se obtendrá una expresión en la que se pueden incluir nuevos conceptos, y es el Terminology Binding el encargado de indicar la tabla del modelo común de datos en la que se han de almacenar dichos conceptos.

4 TECNOLOGÍAS EMPLEADAS

4.1 Java

El lenguaje de programación Java [10] fue desarrollado en 1995 por James Gosling de Sun Microsystems, empresa posteriormente adquirida por Oracle. Se trata de un lenguaje de propósito general, orientado a objetos y basado en clases. Su sintaxis deriva de C y C++, pero elimina herramientas de bajo nivel como la manipulación directa de punteros, con el objetivo de prevenir errores.

Uno de los propósitos de Java es permitir a los desarrolladores implementar programas para ser ejecutados en cualquier dispositivo, ya que una vez que las aplicaciones son compiladas, se pueden ejecutar en cualquier máquina virtual Java (JVM) independientemente de la arquitectura del sistema.

Es uno de los lenguajes de programación más ampliamente utilizados en el mundo, principalmente en aplicaciones web.

La mayoría de los programas desarrollados en los proyectos INTEGRATE y EURECA, están implementados bajo Java, entre ellos la herramienta implementada en el presente Trabajo de Fin de Grado.

Se ha optado por utilizar Java entre otros motivos, por la amplia colección de librerías disponibles para este lenguaje y que son útiles para el desarrollo de esta herramienta, así como para facilitar la interoperabilidad con los diferentes servicios en los que se integra el normalizador semántico.

Entre las librerías utilizadas una de las más importantes es el conector JDBC [11] (Java Database Connectivity API), se trata de un software que permite la interacción entre aplicaciones Java y bases de datos, utilidad básica en el programa a desarrollar, ya que se ha de consultar toda la información presentes en una base de datos y almacenar dicha información normalizada en otra base de datos.

Debido a que también será necesario consultar bases de datos en RDF, se hace uso de las librerías Java RDF Framework de Sesame [12], las cuales permiten la interacción con bases de datos diseñadas según este estándar. En las siguientes subsecciones se presentará información sobre RDF.

4.2 Bases de datos relacionales

4.2.1 MySQL

MySQL [13] es un sistema gestor de bases de datos (SGBD) muy extendido y libre, aunque la mayor parte del copyright del código pertenece a la empresa Oracle.

Se trata de un sistema gestor de bases de datos relacional, multihilo y multiusuario, ha sido desarrollado sobre C y C++.

Las bases de datos que almacenan la información clínica utilizada en los proyectos INTEGRATE y EURECA hacen uso de este gestor, por su estabilidad y facilidad de uso.

4.3 Bases de datos no relacionales

4.3.1 RDF

RDF (Resource Description Framework) [14] es un modelo estándar para el intercambio de datos en la Web. El objetivo es describir la semántica de los datos para ser procesables por máquinas. Entre sus características se encuentran la posibilidad de la fusión de datos basados en esquemas diferentes y permitir la evolución de dichos esquemas sin necesidad de hacer cambio en los sistemas que hacen uso de estos datos.

Este estándar se base en la estrategia de enlazado de la Web basado en URIs (Uniform Resource Identifier), y extiende esta estrategia de nombrado para describir relaciones entre entidades o cosas, las cuales vienen identificadas de igual manera por una URI. El componente producido por dicha relación es conocido como tripleta y está formado por un Sujeto, un Predicado y un Objeto.

El conjunto de estructuras de este tipo forman grafos RDF, y el conjunto de dichos grafos conforman un RDF Dataset, que puede ser consultado utilizando el lenguaje SPARQL, que será descrito en las siguientes subsecciones.

4.3.2 OWL

OWL [15] (Web Ontology Language) es un lenguaje de la Web Semántica avalado por el W3C, diseñado para representar conocimiento exhaustivo y completo sobre entidades, grupos de entidades y las relaciones que puede haber entre ellas.

Los vocabularios médicos descritos en la sección 3.2 del presente documento y utilizados en los proyectos en los que se enmarca este Trabajo de Fin de Grado, están representados bajo este lenguaje y contenido en un archivo OWL accesible mediante conectores rdf y consultados haciendo uso del lenguaje SPARQL.

4.4 Lenguajes de consulta

4.4.1 SQL

El lenguaje de consulta más utilizado en la herramienta normalizadora desarrollada es SQL (Structured Query Language), se trata de un lenguaje declarativo que permite el acceso a bases de datos relacionales.

En los años 70, los laboratorios de IBM definieron el lenguaje de consulta SEQUEL (Structured English Query Language) del que deriva SQL, introducido de manera comercial por Oracle.

SQL permite el cálculo relacional así como el uso del álgebra para elaborar consultas que recuperar o modificar información presente en bases de datos.

Todas las consultas realizadas contra las bases de datos que almacenan la información clínica que ha de ser normalizada por el normalizador semántico se realizan en SQL.

4.4.2 SPARQL

SPARQL [16] es un lenguaje estandarizado de consulta para grafos RDF, normalizado por el RDF Data Access Working Group del W3C. A diferencia de SQL no es utilizado para la inserción de datos, aunque nuevas propuestas intentan añadir esta funcionalidad al lenguaje.

Los resultados de las consultas suelen ser presentados de manera que sean entendidos por personas o por máquinas.

La herramienta de normalización semántica hace uso de SPARQL para realizar consultas a la ontología médica, ya que en algunos casos será necesario recuperar información relacionado con algunos conceptos médicos. Para realizar estas consultas desde el programa Java se usan las librerías RDF Sesame.

5 ESPECIFICACIÓN DE REQUISITOS

5.1 Introducción y Propósito del Sistema

En esta sección se presenta la Especificación de Requisitos Software (ERS), detallando las restricciones en el diseño e implementación del sistema, así como las funcionalidades que deben estar presentes en la herramienta.

Para realizar la ERS se seguirán las especificaciones descritas en el estándar *IEEE Recommended Practice for Software Requirements Specications, IEEE Std. 830-1998* [17]. La organización y el formato dadas por este documento no serán seguidas con completa exhaustividad, ya que la especificación de requisitos va dirigida al tutor del Trabajo de Fin de Grado y los usuarios de las plataformas INTEGRATE y EURECA, por lo que se dará una mayor importancia a las especificaciones que debe proporcionar la herramienta de normalización semántica y a la manera en la que se debe dar uso al sistema.

Los detalles de diseño e implementación no serán incluidas en este apartado, las secciones 6 y 7 de este documento recogerá todo lo relacionado con este aspecto.

5.1.1 Ámbito del Sistema

La herramienta tiene como tarea la normalización semántica de bases de datos que almacenan información de ensayos clínicos. Para ello, se consultan los datos almacenados en una base de datos origen, para ser almacenados en otra base de datos destino, en la que los datos se encuentren representados según mecanismos de normalización semántica proporcionado por terminologías médicas.

La representación en forma normal de algunos datos genera nuevas información, por tanto, el sistema ha de encargarse de crear las instancias necesarias para almacenar dicha información adicional, así como de relacionar esta información con los datos originales.

El objetivo de la generación de las bases de datos normalizadas, es proporcionar a los usuarios finales una recuperación de los datos más completa, homogénea y exhaustiva.

5.1.2 Definiciones, Acrónimos y Abreviaturas

- **HL7.** *Health Level Seven*, es una organización dedicada al desarrollo de estándares que faciliten el intercambio de información clínica de manera electrónica.
- **HL7 v3.** Estándar de mensajería definido por HL7.
- **RIM.** *Reference Information Model*, raíz de todos los modelos de información y estructuras desarrolladas como parte del estándar HL7 v3.
- **OWL.** *Web Ontology Language*, un lenguaje de marcado para publicar y compartir datos usando ontologías en la Web.
- **CDM.** *Common Data Model*, modelo común de datos utilizado en las bases de datos de los proyectos INTEGRATE y EURECA.
- **SNOMED CT.** *Systematized Nomenclature of Medicine – Clinical Terms*, terminología clínica integral, multilingüe y codificada.
- **LOINC.** *Logical Observation Identifiers Names and Codes*, sistema de codificación universal para test, mediciones y observaciones clínicas.
- **HGNC.** *HUGO Gene Nomenclature Committee*, autoridad encargada de asignar una nomenclatura estandarizada de genes humanos.
- **Short normal form de SNOMED.** Mecanismo de representación de conceptos biomédicos codificados en SNOMED que se puede generar aplicando una serie de reglas lógicas sobre una expresión válida.
- **Proyectos INTEGRATE y EURECA.** Proyectos de ámbito europeo que tienen como objetivo el desarrollo de una capa de interoperabilidad para el intercambio de información proveniente de ensayos clínicos.
- **Terminology Binding.** Herramienta presente en los proyectos INTEGRATE y EURECA encargado de asociar conceptos codificados bajo alguna de las terminologías médicas usadas en estos proyectos con las tablas presentes en las bases de datos que toman como modelo el CDM.

- **Data Push.** Servicio proporcionado en la plataforma de los proyectos INTEGRATE y EURECA, encargado de insertar mensajes diseñados según el estándar HL7 v3 en el CDM.
- **Core Dataset.** Servicio proporcionado en la plataforma de los proyectos INTEGRATE y EURECA que provee, entre otras cosas, del mecanismo de normalización de términos en SNOMED.
- **MySQL.** Sistema de gestión de bases de datos relacionales, multihilo y multiusuario proporcionada por Oracle Corporation.
- **JSDK.** *Java Servlet Development Kit*, es un paquete que contiene todas las clases y las interfaces necesarias para desarrollar servlets.

5.2 Descripción Global del Sistema

En esta sección se describirán los factores que afectan al producto final y a sus requisitos específicos.

5.2.1 Perspectiva del Producto

Esta herramienta se ha de integrar dentro de un servicio de inserción de datos presente en las plataformas de INTEGRATE y EURECA. Por tanto, no será necesario incluir una interfaz de usuario, y será proporcionado en forma de librería que pueda ser usada por servicios como el de almacenamiento de datos mencionado anteriormente, utilizando una serie de parámetros de entrada.

A su vez, el sistema también podrá ser usado sin necesidad de integrarse en ningún servicio.

5.2.2 Funciones del Producto

La función principal del sistema es, dada una base de datos basada en el modelo RIM de HL7 que contiene información proveniente de instituciones clínicas en las que se realizan trabajos de investigación biomédica, en las que los conceptos médicos se encuentran codificados según los vocabularios médicos SNOMED CT, LOINC y HGNC, almacenar en otra base de datos diseñada según el mismo modelo que la original, la misma información contenida en esta representada en forma normal.

Debido a que el único vocabulario médico que presenta mecanismos de normalización es SNOMED CT, sólo serán normalizados los conceptos codificados en esta terminología. Los conceptos codificados en LOINC y HGNC serán almacenados sin cambios en la base de datos que resulta del proceso de normalización. Cabe destacar que la gran mayoría de los conceptos presentes en las bases de datos de los proyectos en los que se utilizará esta herramienta están representados en SNOMED CT.

El proceso de normalización también realizará tareas de comprobación de completitud en los datos originales, esto es, se encargará de completar valores de atributos de las

tablas de la base de datos que no han sido proporcionados en los datos originales y que han de llevar una serie de valores por defecto según el tipo de datos que se trate.

5.2.3 Características de los Usuarios

Los usuarios finales del producto será el personal que participa en los proyectos INTEGRATE y EURECA. Debido a que se integra dentro de otros servicios, los usuarios no interactuarán directamente con el sistema, aunque será necesario que tengan conocimiento de los servicios en los que se incluye. Por tanto, deben conocer el funcionamiento del servicio Data Push y la construcción de mensajes según el estándar HL7 v3.

Si el sistema se utiliza de manera independiente, se requiere que el usuario tenga conocimientos en Java para ser capaz de utilizar la librería que contiene la herramienta.

5.2.4 Restricciones

Entre las restricciones del sistema, se encuentra la necesidad de que la base de datos original se base en el CDM, no será posible normalizar bases de datos que no sean generadas según este modelo. Además, los conceptos médicos presentes en dichas bases de datos han de estar codificados bajo las terminologías médicas SNOMED-CT, LOINC y/o HGNC.

A su vez, sólo serán normalizados los conceptos presentes en las tablas Act y Entity del modelo, ya que, por ahora, en los proyectos INTEGRATE y EURECA sólo es necesario tener normalizados actos y entidades.

Por otro lado, es necesario que el sistema en el que se ejecute la herramienta posea un gestor de bases de datos, en particular MySQL.

Debido a que la herramienta está desarrollada bajo Java, es necesario que el sistema en el que se ejecute disponga del JSDK adecuado, en versiones 1.6.0_20 o superiores.

Cabe destacar que, siempre y cuando se cumplan las restricciones anteriores, el sistema podrá ser ejecutado en cualquier sistema operativo.

5.2.5 Suposiciones y Dependencias

Se supone que los usuarios del sistema tienen autorización para tratar los datos médicos almacenados en las bases de datos a normalizar y que harán un uso adecuado de los mismos, asegurando la privacidad de los pacientes que han participado en los ensayos clínicos que han dado como resultado los datos a tratar.

La herramienta depende de la versión del CDM usada, por tanto, actualizar este modelo puede implicar realizar modificaciones del sistema.

5.3 Requisitos específicos

En esta sección se muestra una lista de requisitos funcionales que ha de cumplir el sistema, describiendo todas las funciones que ha de proporcionar la herramienta. Todos estos requisitos son de tipo esencial, por lo que no será aceptable que el sistema no cumpla alguno de ellos.

La lista de requisitos presentada está orientada a los desarrolladores y a los usuarios finales (personal de los proyectos INTEGRATE y EURECA), dando a conocer los objetivos del sistema.

La representación de cada requisito sigue el siguiente formato:

[REQ.#XX]. *Descripción del requisito XX en lenguaje natural*.

Los requisitos a cumplir son los siguientes:

- **[REQ.#01].** El sistema ha de ser presentado como una librería Java, en la que se invoque a una clase principal que recibe como parámetros el nombre de la base de datos a normalizar y un fichero de configuración, por tanto, no se ha de diseñar una interfaz de usuario a alto nivel.
- **[REQ.#02].** La base de datos a normalizar ha de estar diseñada según el CDM de las plataformas proporcionadas en los proyectos INTEGRATE y EURECA.
- **[REQ.#03].** La base de datos a normalizar debe contener información clínica codificada en al menos uno de los vocabularios médicos utilizados en los proyectos INTEGRATE y EURECA. Estos son, SNOMED-CT, LOINC y HGNC.
- **[REQ.#04].** Sólo se normalizarán conceptos codificados en SNOMED-CT, manteniendo sin cambios aquellos codificados en LOINC o HGNC.
- **[REQ.#05].** Sólo se normalizarán conceptos presentes en las tablas Act y Entity del CDM.
- **[REQ.#06].** Se utilizará como mecanismo de normalización Short Normal Form de SNOMED proporcionado por el servicio Core Dataset de los proyectos INTEGRATE y EURECA.
- **[REQ.#07].** El sistema ha de encargarse de dar los valores correctos de aquellos atributos que se encuentren vacíos en las tablas de la base de datos original, de manera que en la base de datos normalizada dispongan de los valores por defecto que indique HL7.
- **[REQ.#08].** El sistema ha de poder integrarse en el servicio Data Push de INTEGRATE y EURECA, de forma que al insertar nuevos datos en una base de datos, sean normalizados e incluidos también en la base de datos normalizada.
- **[REQ.#09].** Sólo podrá ser normalizada una base de datos en cada ejecución del servicio.
- **[REQ.#10].** La base de datos normalizada se identificará con la cadena de texto *normalized_* concatenada con el nombre de la base de datos origen.

- **[REQ.#11].** Se proporcionará la opción, en los casos en los que ya exista la base de datos normalizada previamente, de borrar dicha base de datos o de mantener los datos existentes sin cambios.
- **[REQ.#12].** Las bases de datos, original y normalizada se gestionarán mediante MySQL.
- **[REQ.#13].** El fichero de configuración pasado como parámetro de entrada debe contener la información relacionada con la base de datos a normalizar.
- **[REQ.#14].** Se informará por consola de los posibles fallos que se puedan presentar en la ejecución del sistema.
- **[REQ.#15].** El sistema se limitará a realizar las tareas de normalización y de completar atributos vacíos, no alterará ni la información clínica original ni la de los pacientes.
- **[REQ.#16].** Se proporcionará un archivo log con información tras la terminación del proceso: tiempo de ejecución, número de conceptos normalizados, posibles errores, etc.

6 DISEÑO DEL SISTEMA

El objetivo de esta sección del documento es describir en detalle la manera en que ha sido diseñado el sistema para conseguir un funcionamiento adecuado del mismo una vez implementado.

Cabe destacar que el diseño ha sido realizado de manera que los requisitos presentados en la sección anterior sean satisfechos, teniendo en cuenta las necesidades del sistema, así como las restricciones impuestas por dichos requisitos.

A su vez, se recogen los motivos por los cuales se ha decidido que el diseño del sistema sea el presentado y no otro.

6.1 Aspectos generales del diseño

En esta sección se muestra el diseño general del sistema, mostrando las diferentes tareas que ha de ir realizando el proceso de normalización semántica de una base de datos. Se ha optado por representar el funcionamiento de la herramienta como una serie de pasos que en las subsecciones posteriores serán descritas con detalle, debido a que se trata de un proceso en el que las tareas han de realizarse de manera secuencial, de manera que una tarea no podrá comenzar hasta que haya finalizado la anterior.

Esta forma de representación mostrará con claridad las decisiones de diseño tomadas y será útil para comprender como se realizarán cada una de las tareas que forman parte del proceso completo.

El proceso de normalización semántico de una base de datos que almacena información de ensayos clínicos comienza con una fase de preparación de la base de datos normalizada, en la que se almacenarán datos normalizados de una base de datos origen.

Una vez preparada la base de datos, se lleva a cabo una copia de los datos cuyo valor se mantiene invariable tras la normalización, para luego comenzar con el proceso de normalización del resto de los datos.

Esta última fase, la más importante y compleja del proceso, se divide a su vez en una serie de pasos descritos a continuación.

En primer lugar, se obtiene la forma normal de los conceptos que describen actos de historias clínicas, que pueden ser, observaciones, procedimientos o administraciones de sustancias, y el enlace o binding de la representación en forma normal con el modelo común de datos o CDM.

A continuación, se copia la información relacionada con dichos actos, como puede ser la que establece la relación entre los actos y los pacientes o entre un acto con otro.

Finalmente, se almacena la información normalizada en las tablas correspondientes teniendo en cuenta el enlace obtenido entre los conceptos que forman parte de la forma normal y las tablas del CDM.

En la figura 6.1, se muestra un diagrama con las diferentes fases que forman el proceso de normalización tal y como se ha decidido diseñar.

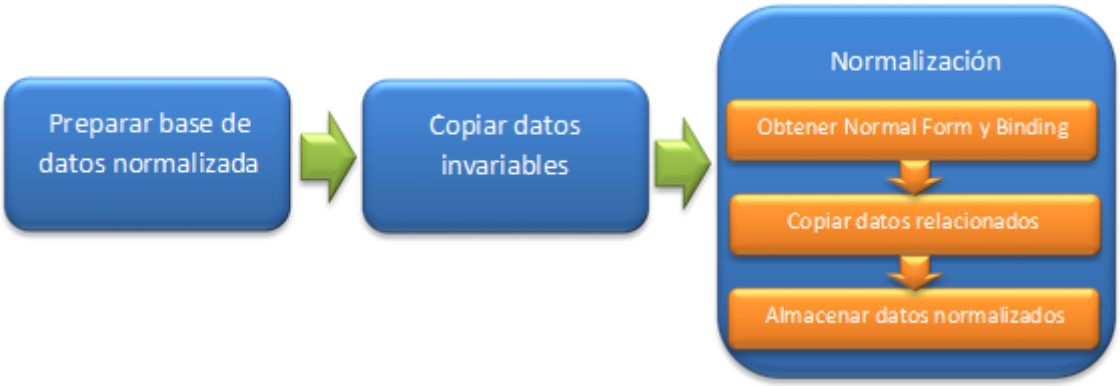


Figura 6.1: Visión general del Proceso de Normalización Semántica

Cabe destacar que en las fases de almacenamiento de datos, se realizan comprobaciones en los valores de algunos atributos presentes en la base de datos origen, para proporcionarles un valor por defecto en caso de que su valor sea vacío.

El proceso de normalización se ha diseñado tomando como elemento central cada uno de los actos clínicos presentes en la base de datos origen, ya que es la manera más sencilla de obtener el resto de información presente en la base de datos, ya sea para su normalización, para completar campos o para realizar una copia de datos que se mantienen sin cambios al final del proceso. Todos estos datos están relacionados de una forma u otra con los actos, por tanto, podemos acceder a ella a través de los mismos

consiguiendo obtener una base de datos resultado en la que los datos se representan en forma normal y no se pierde ningún tipo de información adicional que pueda haber en la base de datos origen.

6.2 Etapas del proceso

En la presente sección se describirá en profundidad las diferentes tareas que lleva a cabo el proceso de Normalización Semántica. Se incluye información que da a entender una serie de decisiones que serán reflejadas en la implementación, la cuál será tratada en mayor profundidad en la sección 7 del presente documento.

6.2.1 Preparar base de datos normalizada

Es la primera tarea de la herramienta normalizadora de bases de datos. En esta primera fase, se prepara la base de datos en la que se almacenará la información normalizada, es decir la base de datos resultado de normalizar una origen cuyo nombre es enviada como parámetro.

Las bases de datos normalizadas mantendrán el mismo nombre que la original añadiendo al principio la cadena de texto *normalized_*. Así por ejemplo, si se normaliza la base de datos *pruebas_normalization*, la base de datos normalizada se identificará por *normalized_pruebas_normalization*.

El siguiente diagrama muestra el funcionamiento de esta tarea:

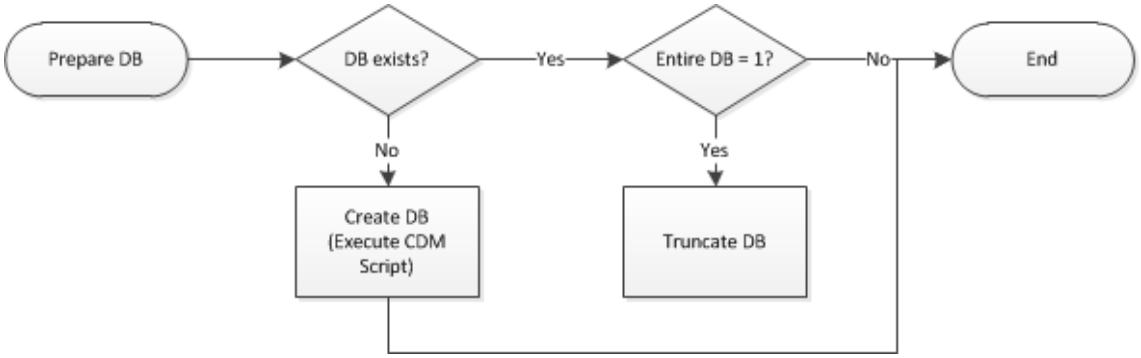


Figura 6.2: Preparar base de datos normalizada

Como se puede observar en la figura 6.2, preparar la base de datos normalizada se trata de una tarea sencilla. En primer lugar, se comprueba si ya existe una base de datos normalizada de la base de datos origen, en caso de que no exista, se ejecutará el script con la versión adecuada del Common Data Model y la preparación de la base de datos habrá finalizado. En el caso de que la base de datos ya exista en el servidor, se comprobará un parámetro *EntireDB*, el cuál valdrá 1 un caso de que se quiera normalizar una base de datos entera, lo que implicará borrar el contenido de la base de datos normalizada. Si no se desea normalizar una base de datos completa el contenido de la base de datos normalizada se mantendrá invariable y se añadirá la nueva

información. Una vez tomada la decisión de borrar o no la información, se termina esta tarea.

El motivo de dar la posibilidad de borrar o no una base de datos creada anteriormente, viene dado por las necesidades del servicio en el que se integrará el sistema. Este servicio se encarga de insertar mensajes HL7 v3 en el CDM, dichos mensajes crearán nuevas instancias en las tablas de la base de datos manteniendo la información previamente cargada, por tanto, al utilizar la herramienta de normalización semántica dentro de este servicio, no se borrará información previamente cargada y normalizada.

Cuando la herramienta es utilizada de manera independiente, suele tener como objetivo normalizar bases de datos enteras, y no sólo parte de los datos. En este caso, si existe previamente una base de datos normalizada, su contenido será borrado para asegurar que se trata de una normalización completa y actualizada de la base de datos origen.

Se ha tomado la decisión de realizar esta tarea al comienzo del proceso ya que es requisito indispensable tener preparada la base de datos que almacenará la información clínica normalizada.

6.2.2 Copiar datos invariables

El Common Data Model almacena información que se mantiene sin cambios tras el proceso de normalización. Dicha información proviene de los datos demográficos de los pacientes y de una tabla utilizada para cachear una serie de conceptos que optimizan las tareas de otros servicios de la plataforma en la que se integra el sistema.

Tras tener preparada la base de datos normalizada, se copia toda esta información en las tablas correspondientes. Como se muestra en la figura 6.3 las tablas que almacenan datos de pacientes son *Person*, *Living Subject*, *Entity*, *Role*, y *Participation*, la tabla utilizada para tareas de optimización es *Cache*.

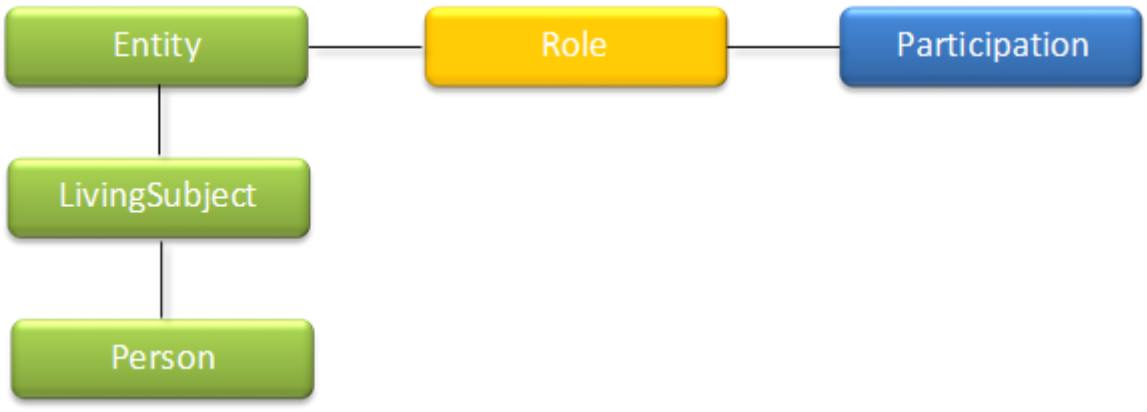


Figura 6.3: Tablas con información de pacientes en CDM

Esta tarea es realizada como segunda etapa del proceso, sin embargo, se podría haber tomado la decisión de desplazarla a la última etapa, pero debido a que es indiferente que

se lleve a cabo antes o después del proceso de normalización ya que no presenta mejoras en eficiencia o rendimiento se ha optado por situarla en la etapa descrita. Además, de esta forma se emplaza en último lugar la etapa de normalización, la cual es la más compleja de todo el proceso.

6.2.3 Normalización

Una vez realizada la fase anterior, comienza la etapa de normalización de conceptos. En ella, se tratarán los actos y datos relacionados de los que se puede obtener una representación en forma normal.

En primer lugar se obtendrá la representación en forma normal, si la tuviesen, de los actos, así como el enlace o binding con las tablas del CDM. En segundo lugar, se almacenará la información relacionada con dichos actos. Finalmente, se almacenarán los datos representados en forma normal en las tablas correspondientes del CDM.

Las diferentes sub-tareas que componen este proceso se describen en más detalle a continuación.

6.2.3.1 Obtener forma normal y binding

La tabla acto mostrada en la figura 6.4, es la principal del Common Data Model, dicha tabla contiene un atributo *code* en el que se almacena un concepto codificado en la terminología médica dada por el valor del atributo *codeVocId* de esta misma tabla.

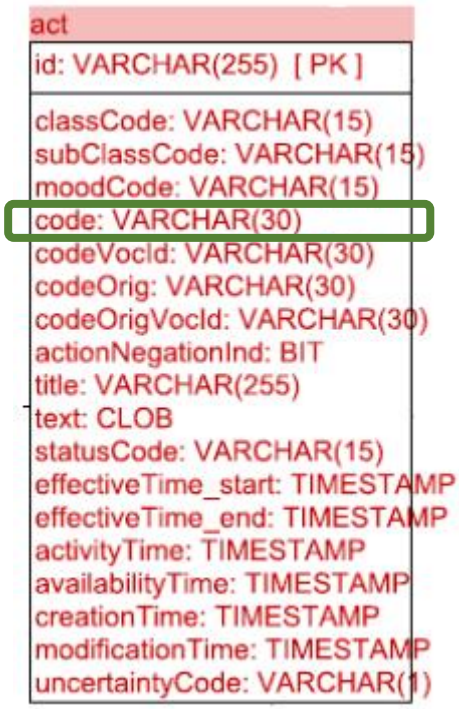


Figura 6.4: Tabla acto del CDM

El proceso de normalización semántica, por medio del servicio Term Binding presente en la plataforma en la que se integra esta herramienta, obtendrá la forma normal de cada uno de los conceptos presentes en el atributo *code* de todas las instancias existentes de actos, así como el enlace de los conceptos que forman parte de la forma normal con las tablas del CDM.

Existen casos en el que el concepto que representa el acto modifique su valor, en estos casos, el concepto original es almacenado en el campo *codeOrig*, y *code* reemplaza su valor por el nuevo.

En la figura 6.5, se puede observar un ejemplo de normalización, en este caso del concepto codificado bajo la terminología SNOMED-CT "5337005|Trocar Biopsy". La representación en forma normal de este concepto produce dos nuevos conceptos, "129314006|Biopsy" y "118418003|Trocar". El Term Binding nos indica que el primero de estos es el método utilizado en el procedimiento de biopsia, y el segundo indica que el dispositivo utilizado para realizar dicha biopsia es un trocar.

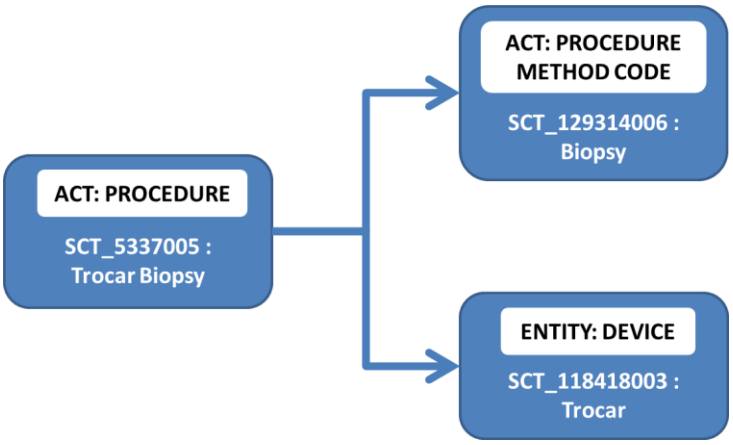


Figura 6.5: Normalización de 5337005|Trocar Biopsy

Cabe destacar, que no todos los actos sufrirán cambios en este proceso, debido a que algunos de ellos ya estarán representados en forma normal en la base de datos origen por ser conceptos primitivos. Por ejemplo, el concepto "367336001|Chemotherapy (procedure)" representado en SNOMED-CT, no generará nueva información ya que su representación en forma normal produce el mismo concepto.

Por otro lado, el proceso de normalización, ha de encargarse de normalizar no sólo los conceptos presentes en la tabla *Entity* del CDM, sino también, los posibles conceptos generados por la normalización de un acto cuyo binding sea la tabla *Entity*, es decir, que se consideren entidades.

6.2.3.2 Copiar datos relacionados

Antes de almacenar los datos generados por la normalización de conceptos presentes en la tabla acto, se ha de copiar la información adicional relacionada con dichos actos. Esta información incluye, la relación existente entre actos, la relación de los pacientes con los actos, y el resto de tablas que pueden contener datos adicionales del acto. La siguiente figura muestra las tablas del Common Data Model en las que habrá que obtener esta información:

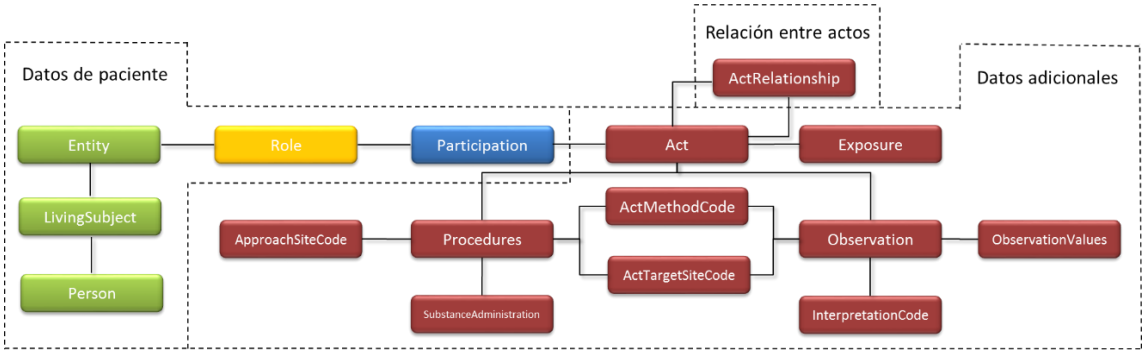


Figura 6.6: Tablas del CDM con datos relacionados con actos

En esta fase del proceso, se llevará a cabo la normalización de los conceptos almacenados en la tabla *Entity* que se relacionan con los actos. Es importante aclarar que la tabla *Entity* también almacena datos relativos a los pacientes que no será modificada. La tarea de normalización de conceptos que representan entidades se lleva a cabo de la misma manera en que se normalizan los actos exceptuando la copia de información relacionada. Es decir, la normalización de entidades implica llevar a cabo una fase de obtención de la forma normal de las mismas y almacenar la posible información generada de la misma forma en la que se almacena la de los actos. En la siguiente subsección se detallará dicho proceso.

Existen una serie de atributos de estas tablas cuyo valor viene determinado por las especificaciones de HL7, en la mayoría de las bases de datos sin normalizar de los proyectos INTEGRATE y EURECA, algunos de estos atributos no tienen asignado ningún valor, es por ello, que el proceso de normalización se encarga de asignarles valores por defecto en el momento de su inserción en la base de datos normalizada. Dichos valores vendrán dados por las reglas de HL7 según el tipo de instancia que se esté almacenando en cada momento.

Al igual que ocurría en la etapa de copia de datos invariables vista anteriormente, esta fase podría realizarse después de la que será explicada en la siguiente subsección, así que se ha optado por tomar la misma decisión y realizarla de manera previa por lo irrelevante de su posicionamiento con respecto a la siguiente etapa.

6.2.3.3 Almacenar datos normalizados

Una vez obtenida la normalización de los diferentes conceptos, se procederá al almacenamiento de los datos normalizados. Para ello, se hará uso del binding obtenido en la fase de obtención de forma normal y binding explicada en la sección 6.2.3.1.

La normalización de un concepto presente en la tabla acto podrá producir nuevas entidades, nuevos métodos utilizados para realizar una observación o llevar a cabo un procedimiento, y nuevos lugares anatómicos que son foco de una observación o un procedimiento. Esto se traduce en nuevas instancias en las tablas *Entity*, *ActMethodCode* o *ActTargetSiteCode* del Common Data Model.

En los casos en los que se generan nuevas entidades, el código del concepto obtenido ha de normalizarse, es decir, habrá que volver a hacer uso del servicio Terminology Binding y comprobar si la normalización de dicha entidad produce nuevos conceptos, los cuáles se almacenan de la misma forma explicada en la presente sección.

Como ocurría en la fase de copia de datos relacionados, habrá que asignar los valores correspondientes al resto de atributos de las tablas, en este caso, los que no son proporcionados por el Terminology Binding.

Una vez almacenados todos los datos invariables y la nueva información normalizada, el proceso de normalización habrá concluido. Se realizará una cuenta de todos los datos que han sufrido normalización y será proporcionada al usuario.

7 IMPLEMENTACIÓN DE LA HERRAMIENTA

En esta sección se describen en detalle las clases y algoritmos utilizados para implementar las diferentes tareas que ha de realizar la herramienta de normalización semántica explicadas en la sección de diseño.

En primer lugar se muestra un diagrama con las diferentes clases que componen el proceso, para luego explicar qué clases se encargan de llevar a cabo las diferentes etapas del proceso y la manera en la que lo hacen.

7.1 Diagrama de clases

La figura 7.1 muestra el diagrama de las clases que constituyen el sistema a desarrollar en este Trabajo de Fin de Grado. Como se puede observar, existe una clase principal *Normalization* en la que se realizan las tareas principales del sistema, y otras secundarias que son instanciadas por la principal, en la etapa del proceso en que sea requerido llevar a cabo las tareas realizadas por dichas clases auxiliares.

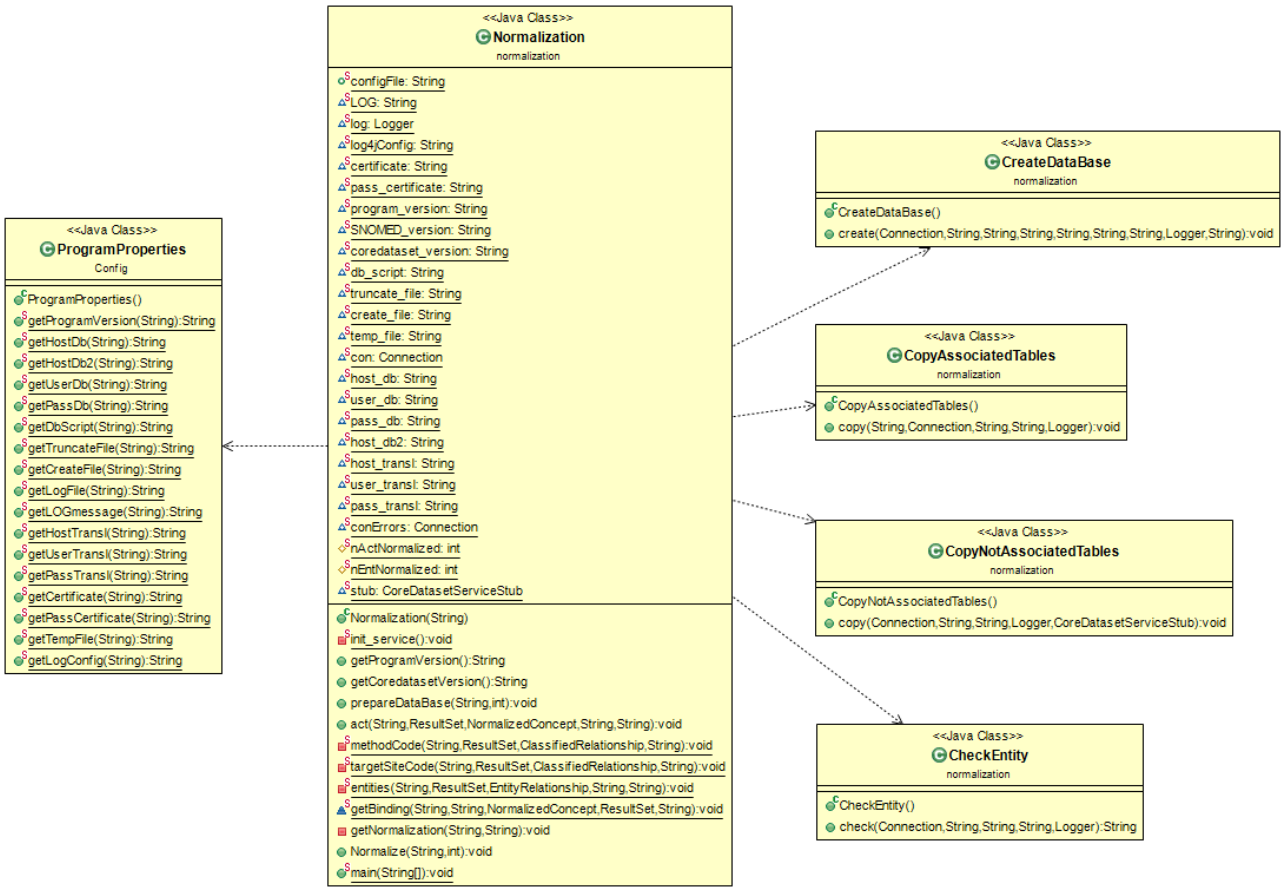


Figura 7.1: Diagrama de clases del proceso de Normalización Semántica

Normalization es el punto de entrada del proceso de normalización, se encarga de realizar las tareas de configuración e inicialización. A su vez, contiene los métodos en los que recae la responsabilidad de proporcionar el correcto funcionamiento de las diferentes etapas que componen el proceso global, coordinando su ejecución y haciendo uso de las clases auxiliares que sean necesarias.

Las clases auxiliares realizan tareas muy específicas que incluso podrían utilizarse de manera independiente, por ello se ha tomado la decisión de separarlas de la clase central. Otro motivo por el que se ha tomado esta decisión, es evitar tener una sola clase sobrecargada de operaciones o métodos.

Entre las clases auxiliares se encuentran *CreateDatabase*, realizando las tareas necesarias para crear una nueva base de datos en el servidor que siga el modelo común de datos de la plataforma, *CopyNotAssociatedTables*, cuya misión es la descrita en la etapa de copia de datos invariables presentada en la sección de diseño, *CopyAssociatedTables*, lleva a cabo la copia de los datos relacionados con los actos que se encuentran en fase de normalización, y *CheckEntity*, operación que comprueba si una entidad producida por la normalización de algún concepto ha sido almacenada con anterioridad en la base de datos, en cuyo caso no se replicará.

Finalmente, la clase *ProgramProperties* se encarga de leer cada uno de los valores necesarios para la configuración del programa leyéndolos de un archivo con extensión *.properties*.

7.2 Descripción de Clases Software

En la presente sección del documento se describirá de manera detallada el funcionamiento de las diferentes clases que implementa la herramienta de normalización semántica. El diagrama de clases de la sección anterior está compuesto por estas clases.

A su vez, para cada una de las clases, se presentarán las decisiones más relevantes que han sido tomadas en la etapa de implementación tomando como referencia el diseño realizado previamente.

Debido a que la clase *ProgramProperties* es usada por el método de inicialización presente en la clase *Normalization*, y a la sencillez de su descripción, ya que sus métodos se encargan de leer valores de un fichero, no se dedicará una subsección a describir esta clase, en su lugar, se detallará su funcionamiento en el apartado dedicado a la clase *Normalization*.

7.2.1 Normalization

Como ya ha sido mencionado, es la clase principal del sistema, el punto de entrada al mismo. Se encarga de las tareas de configuración inicial y de coordinar la secuencia de tareas que ha de ir llevando a cabo el normalizador semántico con el objetivo de obtener el resultado esperado.

La figura 7.2 muestra los diferentes atributos presentes en esta clase, así como los métodos que la componen.

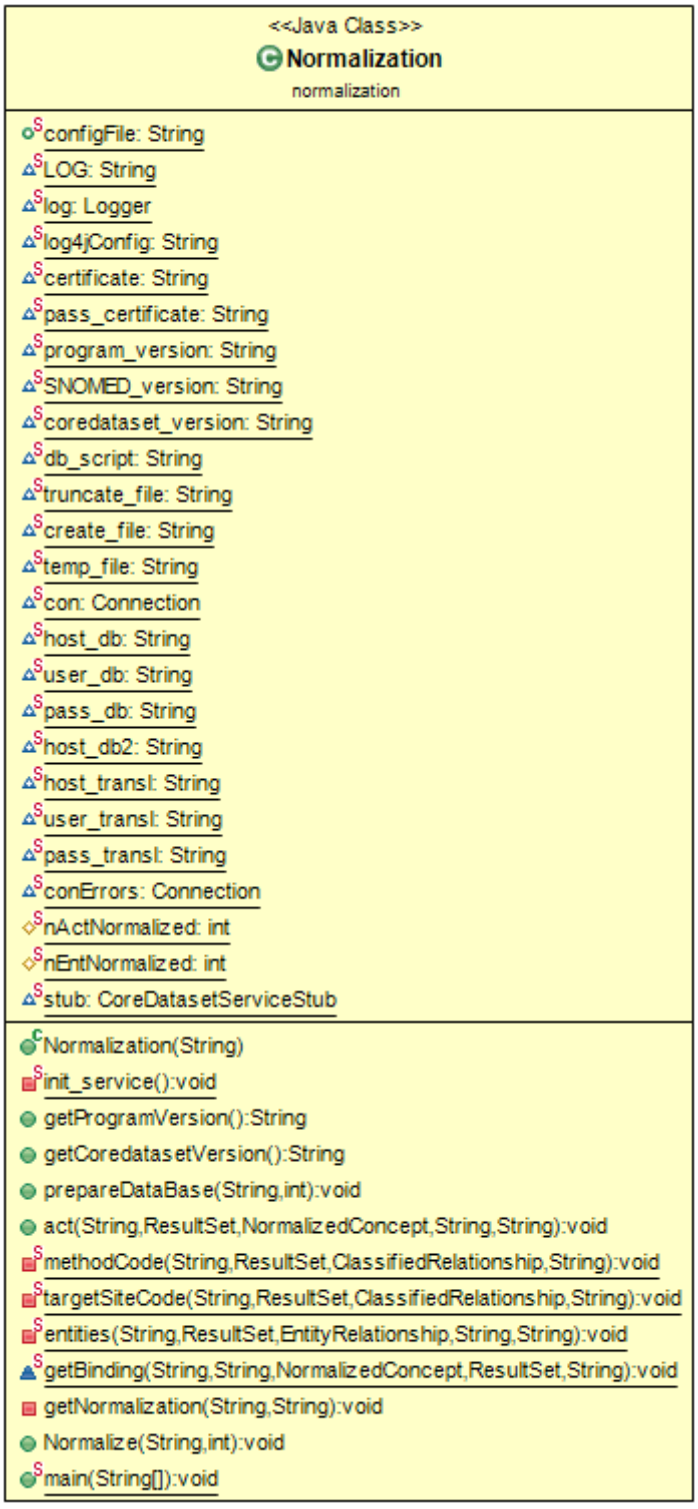


Figura 7.2: Clase Normalization

A continuación se describe la función de cada uno de los atributos:

- **ConfigFile.** Se trata de la ruta del fichero de configuración del programa. Este fichero contiene una serie de valores que se deben asignar al comienzo de la ejecución. En la figura 7.3 se muestra un posible fichero de configuración llamado *propertiesNormalization.config*.

```
program_version=0.3
host_db=jdbc:mysql://localhost:3306/?autoReconnect=true
host_db2=localhost
user_db=testNorm
pass_db=testNorm
db_script=./files/HL7_MySQL_Schema_2.6.3.sql
truncate_file=./files/truncate.sql
create_file=./files/create.bat
temp_file=./files/temp.bat
log4j_config=./config/log4j.properties
LOG=NormalizationPipeline
host_transl=jdbc:mysql://localhost:3306/?autoReconnect=true
user_transl=testNorm
pass_transl=testNorm
```

Figura 7.3: Fichero de configuración de ejemplo

Como se puede observar en la figura anterior, el fichero de configuración asigna a una serie de variables, valores que serán leídos por los métodos presentes en la clase *ProgramProperties* mostrada en la figura 7.4. Dicha clase contiene un método de lectura por cada uno de los valores que han de ser obtenidos a partir del fichero de configuración.

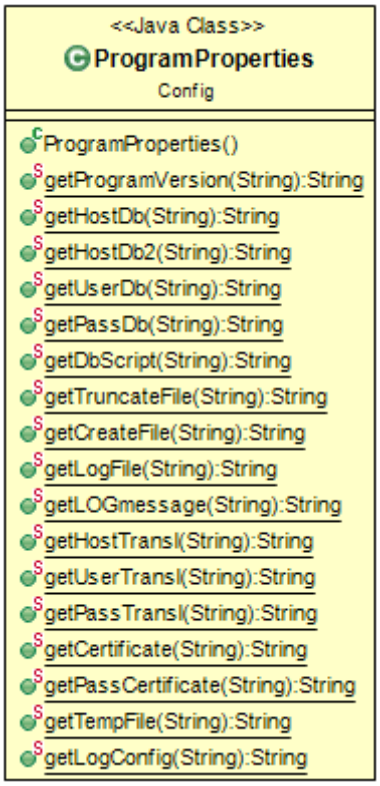


Figura 7.4: Clase ProgramProperties

- **LOG, log y log4jConfig.** Variables utilizadas para configurar y completar el fichero de registro o log. Se utiliza la librería Log4j [18] de Apache Software Foundation [19] para implementar esta funcionalidad.

- **Certificate y pass_certificate.** Necesarias para poder acceder a los servicios desplegados en los servidores de los proyectos INTEGRATE y EURECA. *certificate* almacenará la ruta del certificado de seguridad y *pass_certificate* la contraseña.

- **Program_version, SNOMED_version, coredataset_version.** Versiones de la herramienta de normalización semántica, del vocabulario SNOMED utilizado y el servicio CoreDataset respectivamente.

- **Db_script, truncate_file, create_file y temp_file.** Se trata de rutas a ficheros utilizados en la manipulación de bases de datos. En particular, *db_script* contiene el script del modelo común de datos, *truncate_file* se encarga de vaciar el contenido de la base de datos normalizado si es ejecutado, *create_file* contiene el comando que ha de ser ejecutado para generar una nueva base de datos que

siga el modelo común de datos, finalmente, temp_file es un fichero auxiliar utilizado en la creación de una nueva base de datos.

- **Con, host_db, user_db y pass_db.** *Con* es la conexión a las bases de datos sin normalizar y normalizada, creada según los parámetros de conexión dados por *host_db*, *user_db* y *pass_db* leídos del fichero de configuración.

- **Host_db2.** Contiene el host sobre el que se debe ejecutar el fichero de creación de la base de datos normalizado cuando sea necesario.

- **Host_transl, user_transl, pass_transl y con_errors.** Parámetros de conexión de la base de datos que contiene posibles errores surgidos durante el proceso de normalización, los errores almacenados se producen al intentar almacenar conceptos no presentes en el conjunto de vocabularios utilizados en la plataforma en la que se integra el normalizador. *Con_errors* es la conexión con dicha base de datos.

- **NActNormalized y nEntNormalized.** Contadores del número de actos y entidades que han sido normalizados.

- **Stub.** Permite acceder como cliente a los métodos proporcionados por el servicio *CoreDataset*.

Los métodos proporcionados por la clase *Normalization* son los siguientes:

- **Init_service.** Método de inicialización del programa, se encarga de invocar a los métodos de la clase *ProgramProperties* para obtener los valores de configuración presentes en el fichero *.config*. A su vez, crea la conexión en la base de datos y genera el cliente del servicio *CoreDataset*.

- **Normalize.** Recibe dos parámetros, el nombre de la base de datos a normalizar y la ruta del fichero de configuración. La decisión de no incluir la base de datos a normalizar dentro del fichero de configuración se debe a que, en la plataforma en la que se integra la herramienta, los parámetros contenidos dentro del fichero de configuración en principio deberían cambiar. Sin embargo, dentro de la plataforma puede haber más de una base de datos susceptible de ser normalizada, por ello el nombre de la base de datos a normalizar se pasa como parámetro de entrada.
  Como método de entrada, *normalize*, se encarga de iniciar el proceso de normalización, llamando a otros métodos tanto de la propia clase como de otras externas. En particular, hace una llamada el método *prepareDataBase*, luego llama al método *copy* de la clase *CopyNotAssociatedTables*, y finalmente inicia el proceso de normalización de conceptos invocando al método *getNormalization* de la propia clase *Normalization*.

Tanto los métodos como las clases mencionadas serán descritos más adelante este mismo capítulo de implementación, sin embargo, si es necesario mencionar que esta sucesión de métodos invocados por esta clase no es otra que los pasos explicados en la sección 6.1 de la etapa de diseño, en la que se muestra una visión general del proceso como tres tareas secuenciales, preparar base de datos normalizada, copiar datos invariables y normalización de conceptos.

- **PrepareDataBase.** Realiza la función descrita en la sección 6.2.1 de este documento. En resumen, prepara la base de datos normalizada para comenzar a almacenar datos en ella.

- **GetNormalization.** Invocado por el método *Normalize*, recibe dos parámetros, el nombre de la base de datos origen y el nombre de la base de datos normalizada. Su misión es consultar los conceptos presentes en la base de datos origen, en particular los almacenados en la tabla *Act* del modelo común de datos, para obtener su forma normal.
  Esta tarea se realiza en un bucle que recorre el resultado de consultar todas las instancias de actos presentes en la tabla *Act*, en el que, en primer lugar, se realiza una llamada al método *copy* de la clase *CopyAssociatedTables* encargado de copiar toda la información relacionada con cada acto presente en otras tablas, además de encargarse de normalizar los conceptos que representen entidades también relacionadas con cada uno de los actos. En segundo lugar, se obtiene la forma normal del concepto almacenado en el atributo *code* de la tabla *Act*, realizando una llamada al método TermBinding del servicio CoreDataset. El siguiente paso, es invocar al método *act* de esta misma clase para almacenar correctamente la información del acto, para finalmente obtener el enlace de los nuevos conceptos surgidos de la normalización con las diferentes tablas del modelo común de datos realizando una llamada al método *getBinding*, descrito a continuación.

- **GetBinding.** Método invocado por *GetNormalization*, entre los parámetros que recibe, se encuentra el resultado obtenido tras realizar la llamada al TermBinding con un concepto en particular de la tabla *Act*.
  El resultado de la normalización se encuentra almacenado en un objeto de tipo *NormalizedConcept*, que devuelve los nuevos conceptos que pueden surgir de representar un término en su forma normal indicando en que tabla se han de almacenar.
  Es importante destacar, que sólo se generarán conceptos que se almacenarán en de las tablas *actTargetSiteCode*, *actMethodCode* y *Entity*.
  En líneas generales, el método *getBinding* realiza la función de comprobar a través del objeto de la clase *NormalizedConcept*, qué nuevos conceptos se han generado, es decir, si hay que almacenar nuevas instancias en *actTargetSiteCode*, *actMethodCode* y/o *Entity*, en el caso de que sea necesario,

se realiza una llamada a los métodos *targetSiteCode*, *methodCode* o *entities* de la clase *Normalization* según corresponda.
El siguiente diagrama ilustra el comportamiento de este método:

Figura 7.5: Funcionamiento del método GetBinding

Como se puede observar en la figura 7.5, el método primero comprueba si hay nuevos conceptos que representan métodos de procedimientos o *methods*, y mientras haya en el objeto devuelto por el Term Binding se invoca el método *methodCode* que almacena los nuevos conceptos en la tabla *actMethodCode*. En segundo lugar se realiza la misma comprobación para lugares de la anatomía del paciente en el que se localiza un acto o *target sites*, para finalmente repetir el proceso para entidades o *entities*. Finalmente se retorna de este método.

- **MethodCode y targetSiteCode.** Estos métodos realizan la misma tarea, almacenar los nuevas instancias de las tablas *actMethodCode* y *actTargetSiteCode* respectivamente surgidos tras la normalización. Como ya se ha mencionado anteriormente, son métodos invocados por *GetBinding*.

- **Entities.** Realiza la misma función que los dos métodos anteriores, pero en este caso se almacenan nuevas instancias en la tabla *Entity*. Además, se encarga de normalizar los conceptos que representan las propias entidades que almacena, invocando al Term Binding y al método *getBinding*.

7.2.2 CreateDataBase

Esta clase está compuesta por un único método invocado desde la clase *Normalization*, en concreto por el método *prepareDataBase*. Esto suceda en la primera etapa del proceso de normalización semántica tras haberse inicializado el programa leyendo el fichero de configuración explicado en la sección anterior.

El uso de esta clase depende de una comprobación realizada dentro de *prepareDataBase*, dicha comprobación no es otra que examinar si ya existe la base de datos normalizada de antemano, y será exclusivamente cuando no exista cuando se realice una llamada al método que compone la clase.

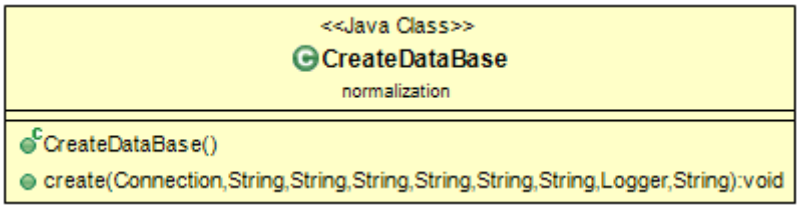


Figura 7.6: Clase CreateDataBase

Como se puede observar en la figura 7.6, el método presente en la clase *CreateDataBase*, se identifica por el nombre *create*. Entre los parámetros más relevantes que recibe se encuentran la conexión MySql y el nombre de la base de datos a crear. Otros de los parámetros importantes se describen a continuación:

- **Ruta del fichero *exec*.** El contenido del fichero exec es el mostrado en la figura 7.7. Se trata del comando MySql a ejecutar para crear una base de datos usando el esquema del modelo común de datos.

```
mysql --user=user1 --password=password1 --host=host1 database1 < .\files\HL7_MySQL_Schema_2.6.3.sql
```

Figura 7.7: Contenido del fichero exec

  Los valores de usuario, contraseña, host y base de datos, toman valores genéricos en este fichero, que serán modificados y almacenados en otro fichero por el propio método *create*. Como se observa en la figura anterior, se ha de incluir la ruta al esquema del modelo común de datos, en este caso se encuentra dentro del fichero de ejecución del programa en la carpeta *files*, y la versión del modelo común de datos es la 2.6.3.

- **Usuario, contraseña, host y base de datos**. Como se ha explicado anteriormente, hay que sustituir los valores de conexión en el fichero *exec*. Dichos valores son enviados como parámetro al método *create*. A su vez, dichos valores son asignados en la inicialización de la herramienta, y por tanto leídos del fichero de configuración. Se ha tomado la decisión de añadir estos valores a dicho fichero para permitir una mayor flexibilidad a la hora de ejecutar en normalizador semántico en diferentes sistemas.

- **Ruta del fichero temp**. Finalmente, el método *create*, recibe a su vez un fichero temporal. Dicho fichero ha de estar creado previamente con permisos de ejecución, lectura y escritura. Su contenido inicialmente es indiferente, ya que se trata de un fichero auxiliar cuyo contenido es borrado antes de escribir en él. Se hace uso de este fichero ya que en él se almacenará el comando mysql con los parámetros de conexión sustituidos por los recibidos como parámetro y explicados en el punto anterior. El comando que finalmente se ejecuta en este

método es el contenido en este fichero, creando las diferentes tablas presentes en el modelo común de datos preparadas para almacenar el contenido normalizado.

El funcionamiento del método *create* está representado en la figura 7.8.

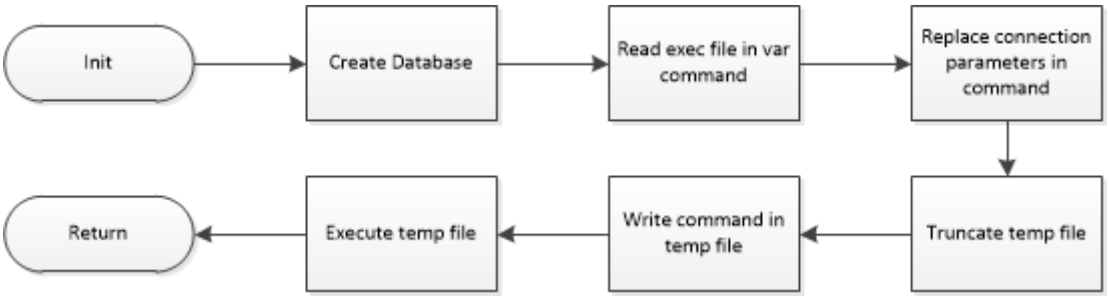


Figura 7.8: Método create

La primera acción realizada por este método es la de ejecutar el comando SQL de creación de una nueva base de datos. En segundo lugar se lee el contenido del fichero *exec*, asignando dicho contenido a una variable *command*, en la que luego se reemplazan los parámetros de conexión genéricos por los presentes en el archivo de configuración del programa. A continuación, se vacía el contenido del fichero temporal para luego escribir en él el valor de la variable *command* con los parámetros de conexión correctos. Finalmente se ejecuta el comando presente en el fichero temporal y se retorna.

7.2.3 CopyAssociatedTables

Esta clase, compuesta por el método *copy*, tiene la tarea de copiar toda la información relacionada con un acto en particular que se encuentre almacenada en otras tablas en la base de datos normalizada.

Es invocado desde el método *getNormalization* de la clase *Normalization*, en el momento de recorrer todos las instancias de actos presentes en la tabla *Act* para llevar a cabo la normalización del campo *code*. De esta manera, no sólo se almacena el acto normalizado sino también el resto de información relacionada con el mismo.

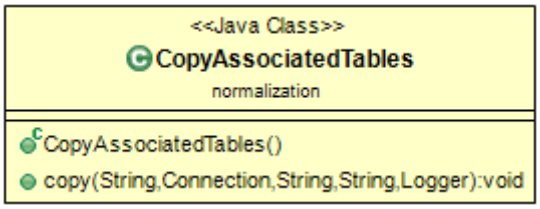


Figura 7.8: Clase CopyAssociatedTables

Las tablas de las cuales se copiarán datos relacionados con actos ya han sido detalladas en el apartado de diseño, por tanto no se incidirá en ese aspecto en esta sección.

Únicamente mencionar, que la copia de las tablas en general se ha realizado haciendo uso de un comando SQL en el que se combina un *INSERT* sobre la base de datos normalizada con un *SELECT* sobre la base de datos origen, tomando siempre como referencia el identificador del acto tratado en cada iteración.

Se realiza una llamada al método *copy* por cada acto presente en la base de datos origen, tanto si presenta forma normal como si no, exceptuando los casos en los que el concepto que represente el acto no pertenezca a las terminología presentes en el servicio *CoreDataset*, en cuyo caso no se almacena el acto en la base de datos normalizada ni la información relacionado que presente, y se informa al usuario del suceso por medio de un aviso o *warning*.

7.2.4 CopyNotAssociatedTables

Al igual que la clase *CopyAssociateTables*, esta clase sólo la compone un método *copy*, cuyo funcionamiento es similar al de la clase anterior. La diferencia radica en las tablas que son copiadas de una base de datos a otra, en este caso, se busca información en las tablas en las que puede haber información que no guarda relación con ningún acto.

Esta tarea, invocado por el método *Normalize* de la clase Normalization tras la preparación de la base de datos normalizada, es necesaria ya que sin su ejecución se podría perder información tras la normalización, y el objetivo final del proceso de normalización semántica es obtener datos normalizados sin pérdida de información.

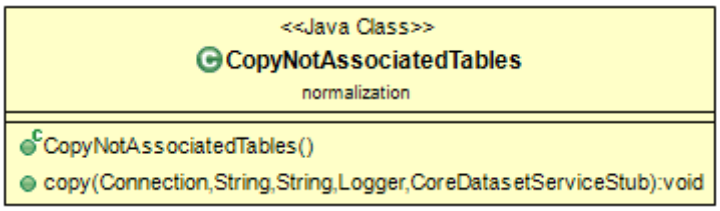


Figura 7.9: Clase CopyNotAssociatedTables

7.2.5 CheckEntity

Las instancias almacenadas en la tabla *Entity* del modelo común de datos, representan entidades, que pueden ser desde personas hasta compuestos farmacológicos. En los casos en los que queramos almacenar una nueva entidad surgida por la normalización de algún concepto, habrá que comprobar si existe ya una instancia que represente dicha entidad, en cuyo caso, no habrá que crear una entrada nueva en la tabla *Entity*, pero sí en las tablas *Role* y *Participation* que establezcan la relación entre dicha entidad y el acto tratado en el momento de ejecución actual.

El cometido del método presente en la clase *CheckEntity* es precisamente el de realizar las comprobaciones previas a almacenar una nueva entidad.

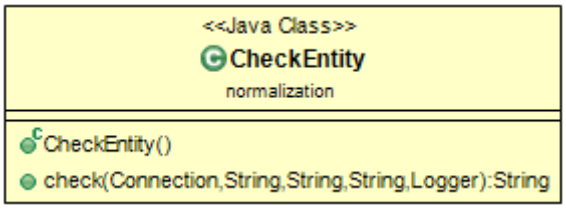


Figura 7.10: Clase CheckEntity

Como se puede observar en la figura 7.10, la clase *CheckEntity* sólo contiene un método *check*. Dicho método realiza una consulta a la tabla *Entidy* de la base de datos normalizada en busca de una entidad cuyo campo *code* coincida con un valor que forma parte de los parámetros de entrada. En el caso de que exista una entidad almacenada que represente el mismo concepto, se devuelve el identificador de la entidad ya existente.

Las llamadas *check* las produce el método *entities* de la clase *Normalization*, ya que es el encargado de almacenar nuevas entidades producidas por la representación en forma normal de algún concepto. El método *entities* es el que se encarga de tomar las decisiones oportunas según el valor devuelto por *check*, de manera que si el valor devuelto es *null*, se crea una nueva entrada tanto en la tabla *Entity*, como en *Role* y *Participation*, y si por el contrario, el valor es un identificador de una entidad, las entradas sólo son creadas en las tablas *Role* y *Participation*.

El motivo por el cual *check* devuelve el identificador de la entidad si existe previamente, es permitir conocer al método *entities*, con qué instancia de *Entity* ha de generar la relación con el acto tratado a través de las tablas *Role* y *Participation*.

8 PRUEBAS Y RESULTADOS

En la presente sección se recogen las diferentes pruebas realizadas para verificar el correcto funcionamiento de la herramienta.

En primer lugar, se llevan a cabo una serie de pruebas individuales que permitan probar el correcto funcionamiento del mecanismo de representación en forma normal.

En segundo lugar, se realizan pruebas sobre bases de datos completas, algunas de ellas almacenan datos reales provenientes de ensayos clínicos, de las que por motivos de confidencialidad no se mostrarán datos ni capturas de las mismas, se mostrará únicamente la comprobación en número de instancias que deben encontrarse en la base de datos normalizada.

Finalmente, se realizan una serie de pruebas de integración con otros servicios de las plataformas INTEGRATE y EURECA, en particular, los servicios de inserción y consulta de datos.

8.1 Pruebas individuales

Para llevar a cabo este conjunto de pruebas, en primer lugar se seleccionan una serie de conceptos en SNOMED-CT que representen actos o entidades. El siguiente paso consistirá en almacenar una serie de instancias en las tablas Act o Entity de una base de datos esquematizada según el modelo común de datos que vengan representadas por los conceptos seleccionando. En dichas instancias, se asignará valor exclusivamente a los atributos necesarios, es decir, a los que no pueden ser vacíos y a los necesarios para llevar a cabo la normalización. El motivo por el que se obvian por ahora el resto de atributos se debe a que en estas pruebas se comprueba exclusivamente el correcto funcionamiento del mecanismo de normalización.

Los conceptos a normalizar son los que se muestran en la siguiente tabla:

Concept	Title	CDM table
373377004	Moderately differentiated histological grade finding	Act
234262008	Excision of axillary lymph node	Act
104934005	Sodium measurement, serum	Act
118418003	Trocar device	Entity
349848009	Epirubicin	Entiy

La primera columna de la tabla anterior, muestra el código del concepto descrito por la segunda columna. La tabla del modelo común de datos en la que se almacena el tipo de concepto viene dada por la tercera columna.

Se han seleccionado un número reducido de conceptos, en los que encontramos representaciones en forma normal diversas. En algunos casos no se debe generar nuevos conceptos, mientras que en otros el resultado de la ejecución de la herramienta debe de producir nuevas instancias en otras tablas.

Debido al gran tamaño de la ontología SNOMED-CT, de aproximadamente trescientos mil conceptos, se ha tomado una muestra pequeña, debido a la imposibilidad de probar con todos los conceptos. Por ello, los conceptos elegidos muestran las diferentes situaciones que pueden producirse. Cabe destacar, que en las pruebas en las que se normalizan bases de datos completas y reales, se normaliza una gran cantidad de conceptos en los que se observó que su normalización era correcta.

La manera de presentar estas pruebas será, en primer lugar, mostrar el resultado esperado de cada una de las pruebas individuales. Dicho resultado es representado como se muestra en el siguiente ejemplo:

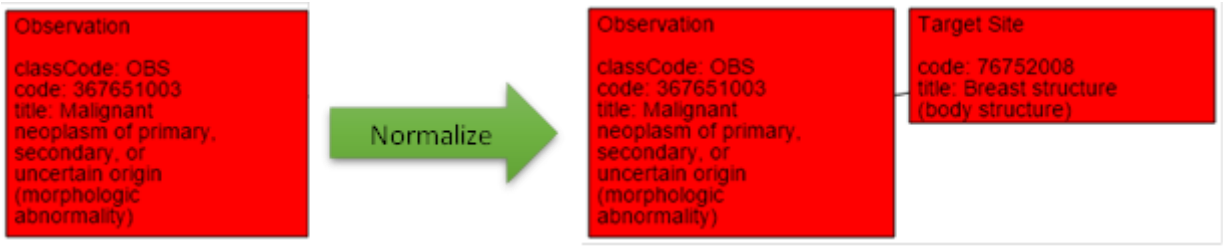


Figura 8.1: Representación del resultado esperado en pruebas individuales

La figura 8.1. muestra el resultado esperado de normalizar el concepto *367651003|Malignant Neoplasm of primary, secondary, or uncertain origin*. En este caso se trata de una observación, recordemos que las observaciones y procedimientos son almacenados en la tabla *Act*, que mantiene inalterable la instancia de la tabla *Act* y que a su vez genera una nueva instancia en la tabla *ActTargetSite* representada por el código del concepto en SNOMED *76752008|Breast structure*.

La manera de demostrar que se ha obtenido el resultado esperado es mostrando capturas de pantalla de las tablas de la base de datos de prueba, de manera que se observe la creación de nuevas instancias en los casos en que esto deba ocurrir.

A continuación se muestran los resultados de las pruebas individuales representados de la manera explicada anteriormente:

- **Observation 373377004|Moderately differentiated histological grade finding**
  - Resultado esperado:

Figura 8.2: Normalización del concepto 373377004

  - Resultado obtenido:

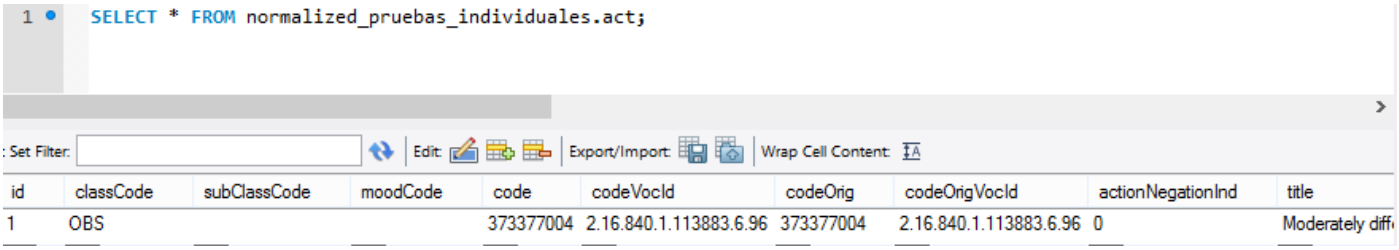


Figura 8.3: Tabla Act base de datos normalizada tras la primera prueba

Como se puede observar en la figura 8.3. el contenido de la tabla *Act* de la base de datos resultado de normalizar otra base de datos origen que contiene el concepto 373377004. Debido a que la representación en forma normal de dicho concepto no produce modificaciones ni añade nuevos conceptos, el atributo *code* se mantiene inalterable en la tabla *Act*.

Cabe destacar que esta primera prueba ha servido para comprobar el correcto funcionamiento de la creación de una base de datos que almacene el contenido normalizado.

- **Procedure 234262008|Excision of axillary lymph**
  - Resultado esperado:

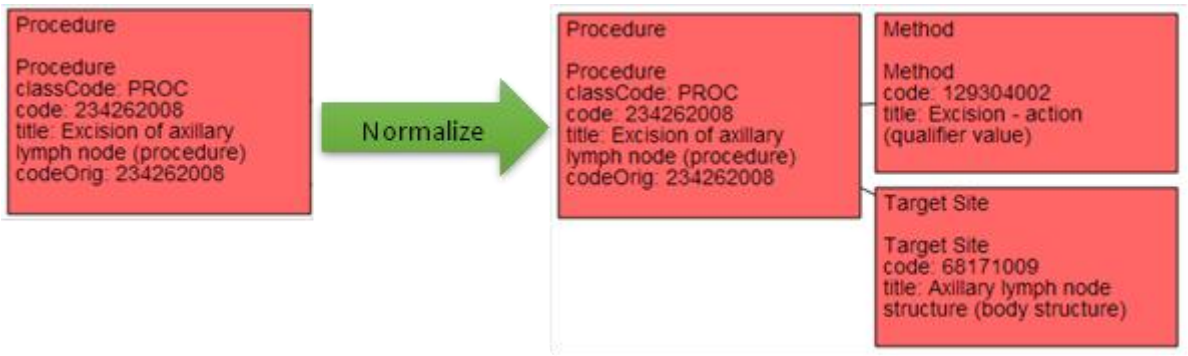


Figura 8.4: Normalización del concepto 373377004

  - Resultado obtenido:

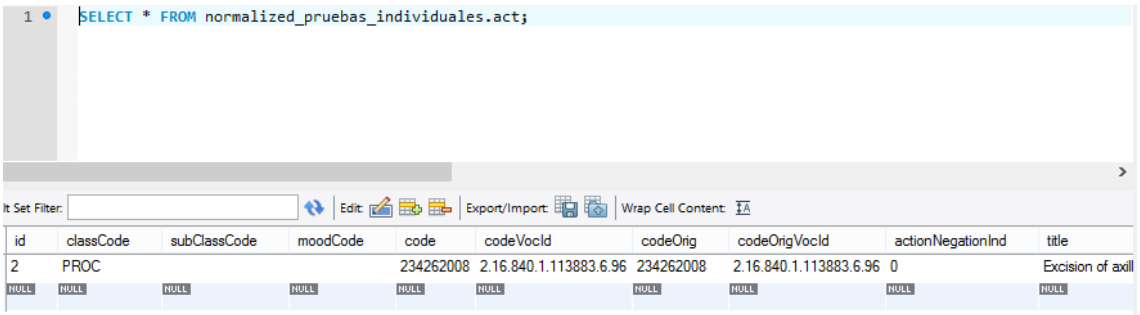


Figura 8.5: Tabla Act base de datos normalizada tras la segunda prueba

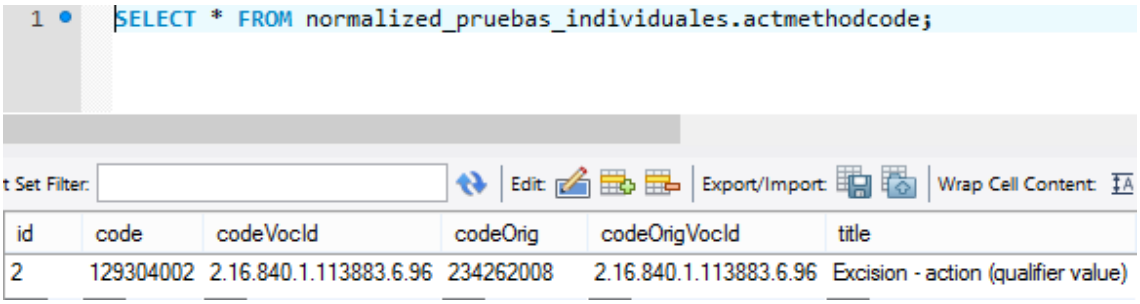


Figura 8.6: Tabla actMethodCode base de datos normalizada tras la segunda prueba

```
1 •  SELECT * FROM normalized_pruebas_individuales.acttargetsitecode;
```

Set Filter: | Edit: | Export/Import: | Wrap Cell Content:

id	code	codeVocId	codeOrig	codeOrigVocId	title
2	68171009	2.16.840.1.113883.6.96	234262008	2.16.840.1.113883.6.96	Axillary lymph node structure (body structure)

Figura 8.7: Tabla actTargetSiteCode base de datos normalizada tras la segunda prueba

El resultado de normalizar el concepto 234262008, produce dos nuevos conceptos, 129304002 y 68171009, que han de ser almacenados en las tablas *ActMethodCode* y *ActTargetSiteCode* respectivamente. El correcto funcionamiento lo demuestran las capturas de pantalla recogidas en las figuras 8.6. y 8.7. que muestran que el contenido de las tablas es el esperado.

- **Observation 104934005|Sodium measurement, serum**
  - Resultado esperado:

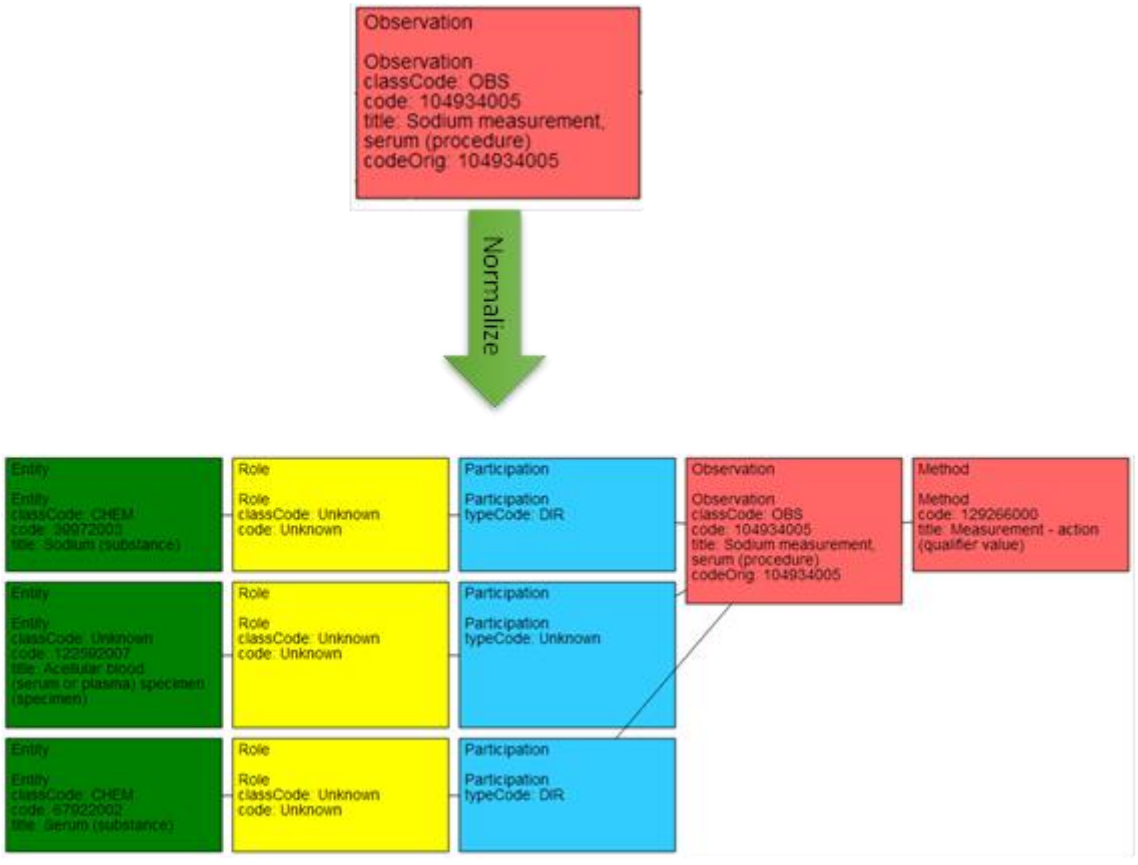


Figura 8.8: Normalización del concepto 104934005

  - Resultado obtenido:

```
1 • SELECT * FROM normalized_pruebas_individuales.act;
```

id	classCode	subClassCode	moodCode	code	codeVocId	codeOrig	codeOrigVocId	actionNegationInd	title
3	OBS			104934005	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	0	Sodium measu

Figura 8.9: Tabla Act base de datos normalizada tras la tercera prueba

```
1 • SELECT * FROM normalized_pruebas_individuales.entity;
```

id	classCode	determinerCode	code	codeVocId	codeOrig	codeOrigVocId	title
048d507c-b8be-483c-9774-88f59e418d41	CHEM	KIND	39972003	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	Sodium (substance)
72a93d7b-9751-4974-b262-2b387aa5cd70	CHEM	KIND	32457005	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	Body fluid (substance)
84cf6fe0f2a5-41e9-8fb7f170bb50b91d	CHEM	KIND	261226001	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	Blood component (substance)
e70af2cd-1e48-4d83-87e4-daaded92af0c	CHEM	KIND	67922002	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	Serum (substance)
ee0321e7-69db-4a82-ba66-565ded57f202	-	KIND	122592007	2.16.840.1.113883.6.96	104934005	2.16.840.1.113883.6.96	Acellular blood (serum or plasma) specimen (specimen)

Figura 8.10: Tabla entity base de datos normalizada tras la tercera prueba

En este caso se generan correctamente las entidades que ha de producir la normalización del concepto 104934005. A su vez se comprueba que se produce normalización recursiva en las entidades producidas, ya que los conceptos *32457005|Body fluid* y *261226001|Blood Component* son resultado de normalizar las entidades producidas por la normalización del concepto original de manera recursiva.

- **Entity 118418003|Trocar device**
  - Resultado esperado:

Figura 8.11: Normalización del concepto 118418003

  - Resultado obtenido:

```
1 • SELECT * FROM normalized_pruebas_individuales.entity;
```

id	classCode	determinerCode	code	codeVocId	codeOrig	codeOrigVocId	title	quantity	name	desc
cd0d9530-555e-4a48-92ef-6693de20c843	DEV	KIND	118418003	2.16.840.1.113883.6.96	5337005	2.16.840.1.113883.6.96	Trocar device (physical object)			NULL

Figura 8.12: Tabla entity base de datos normalizada tras la cuarta prueba

Nos encontramos ante otro caso en el que la normalización no produce cambios.

- **Entity 349848009|Epirubicin**
  - Resultado esperado:

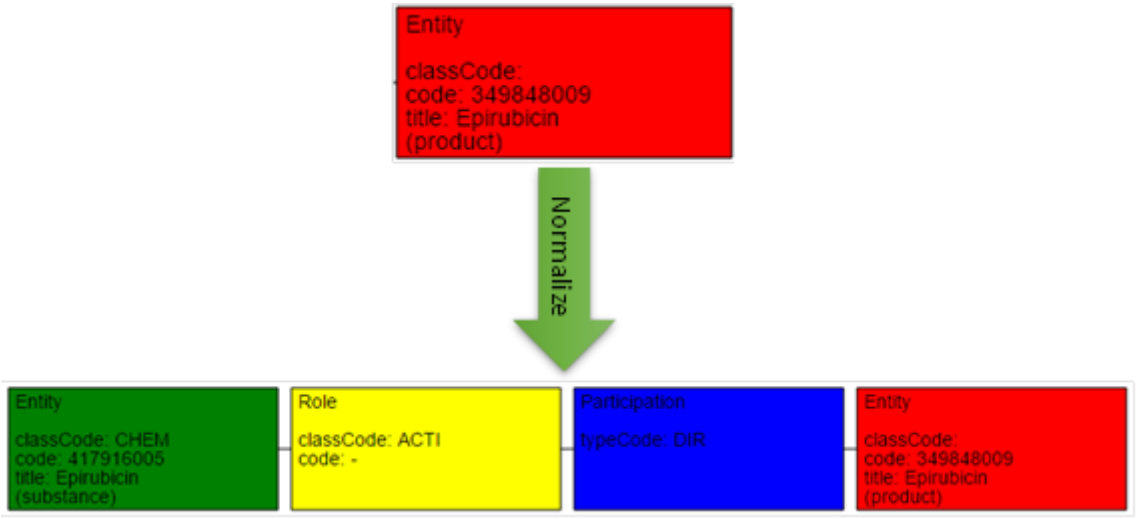


Figura 8.13: Normalización del concepto 349848009

  - Resultado obtenido:

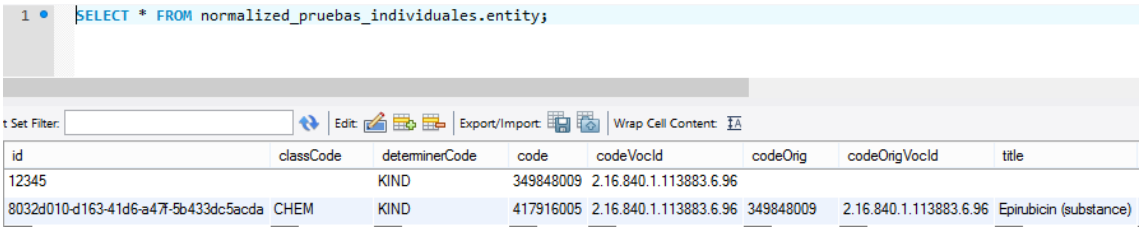


Figura 8.14: Tabla entity base de datos normalizada tras la quinta prueba

En esta última prueba individual comprobamos el correcto funcionamiento de la herramienta a la hora de normalizar entidades que producen nuevas entidades.

8.2 Normalización de bases de datos reales

En esta fase de pruebas de la herramienta de normalización semántica de bases de datos, se examina el funcionamiento de la misma aplicando el programa sobre bases de datos que contienen datos reales provenientes de ensayos clínicos.

Cada prueba se identifica por un número y se presenta en forma de tablas la comparación entre el número de instancias en las bases de datos sin normalizar, con el número de instancias presentes en presentes en las bases de datos normalizadas.

Es de esperar que el número de instancias crezca exclusivamente en las tablas *ActTargetSiteCode*, *ActMethodCode* y *Entity*, ya que la representación en forma normal sólo podrá generar conceptos cuyo enlace se produzca con estas tablas.

- **Prueba #1.**

	NON-NORMALIZED CDM	NORMALIZED CDM
Act code	6413	6413
Observation code	5237	5237
Procedure code	492	492
Substance Administration code	588	588
Entity code	103	112
Target site code	492	2348
Method code	142	1021

En esta primera base de datos encontramos un total de 6413 actos, de los cuales 5237 son observaciones, 492 son procedimientos, 588 administraciones de sustancias, y el resto identifican ficheros, imágenes o el propio identificador de ensayo clínico. Este último tipo de actos se mantienen inalterables tras la normalización.
Como se puede observar en la tabla anterior, el número de actos es el mismo en ambas bases de datos, lo que nos indica que no hemos perdido información, por tanto, la copia de actos invariables ha funcionado correctamente.
Por otro lado, se puede apreciar un incremento sustancial en las instancias de las tablas *ActTargetSiteCode* y *ActMethodCode* como cabría esperar. A su vez, la tabla *Entity* sufre un incremento menor, hecho también esperable, debido a que sólo existe una instancia de entidad por cada concepto, así, una entidad se puede relacionar con un acto o con varios. Sin embargo, una instancia de *ActMethodCode* y *ActTargetSiteCode* sólo se relación con un acto.

- **Prueba #2**

	NON-NORMALIZED CDM	NORMALIZED CDM
Act code	415	415
Observation code	321	321
Procedure code	60	60
Substance Administration code	30	30
Entity code	80	125
Target site code	152	152
Method code	0	157

En esta segunda prueba, se normaliza una base de datos con un número menor de actos, 415, de los cuales 321 son observaciones, 60 son procedimientos, 30 son administraciones de sustancia y 4 son de otro tipo.
La diferencia de esta base de datos con la anterior, es que ya contiene actos que han sido almacenados de acuerdo a su forma normal, en particular, actos que su forma normal incluye conceptos a almacenar en la tabla *ActTargetSiteCode*.
Como se puede observar en la tabla resultado, el número de instancias de esta

última tabla se mantiene invariable, ya que todos los conceptos presentes en la tabla *Act* que generan instancias en la tabla *ActTargetSiteCode* ya presentaban dichas instancias en la base de datos original. De esta forma se comprueba que la herramienta no sobrescribe datos ya normalizados.
Por otro lado, si se observa, al igual que en la primera tabla, un aumento en las instancias de las tablas *ActMethodCode* y *Entity*.

8.3 Pruebas de integración

Para concluir la etapa de pruebas del normalizador semántico, se han realizado dos tipos de pruebas de integración. Una de integración dentro del servicio que hará uso de la misma dentro de las plataformas de los proyectos *INTEGRATE* y *EURECA*, y otra prueba de consultas contra la base de datos normalizada haciendo uso del servicio de consulta proporcionado en los proyectos anteriores.

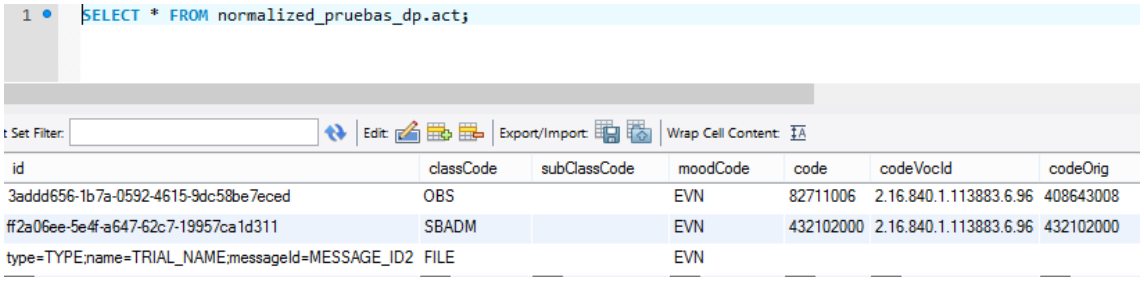
- **Integración Data Push Service.**
  Como se ha mencionado en la sección 3.1 de este documento, la herramienta se integra dentro del servicio Data Push de la plataforma en la que se enmarca este Trabajo de Fin de Grado. El funcionamiento esperado de este servicio es, dado un mensaje HL7 v3, se almacene en una base de datos que luego, a través de la herramienta desarrollada, será normalizada y almacenada en otra base de datos.
  Se llevaron a cabo una serie de pruebas con éxito en los resultados. Un ejemplo de este tipo de pruebas es el mostrado a continuación.
  - **Inserción de un mensaje HL7 a través del servicio Data Push (Normalizador Semántico integrado)**
    Se pretende almacenar el siguiente mensaje HL7 v3 (se muestra el fragmento más relevante del mismo):

```
<section>
<entry>
    <observation classCode="OBS" moodCode="EVN">
        <code code="408643008" codeSystem="2.16.840.1.113883.6.96" codeSystemName="SNOMED CT" displayName="Infiltrating duct carcinoma of breast"/>
        <effectiveTime value="20141218"/>
    </observation>
</entry>
<entry>
    <substanceAdministration classCode="SBADM" moodCode="EVN">
        <code code="432102000" codeSystem="2.16.840.1.113883.6.96" codeSystemName="SNOMED CT" displayName="Administration of substance"/>
        <doseQuantity value="20" unit="mg"/>
        <consumable>
            <manufacturedProduct>
                <manufacturedLabeledDrug>
                    <code code="373345002" displayName="Tamoxifen" codeSystem="2.16.840.1.113883.6.96"/>
                </manufacturedLabeledDrug>
            </manufacturedProduct>
        </consumable>
        <effectiveTime value="20141218"/>
    </substanceAdministration>
</entry>
</section>
```

Figura 8.15: Mensaje HL7 v3 de prueba

Dicho mensaje almacena dos actos, uno de ellos de tipo observación, y otro de tipo administración de sustancia. En el mensaje sólo se incluye el código del acto sin normalizar.

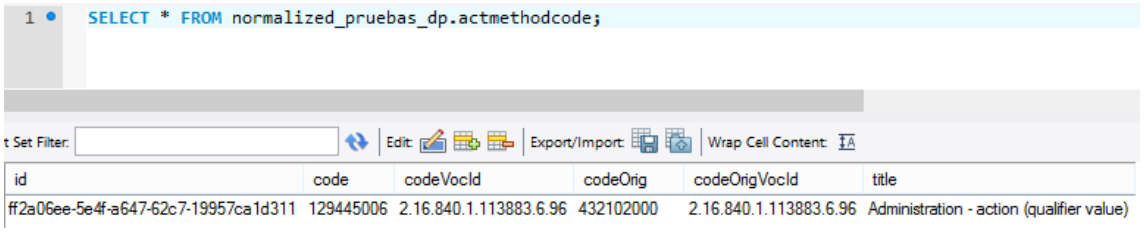
Tras la ejecución de un cliente del servicio Data Push se obtiene una base de datos sin normalizar y otra normalizada por la herramienta desarrollada. El contenido de las tablas de esta última base de datos se muestra en las siguientes figuras.



```
SELECT * FROM normalized_pruebas_dp.act;
```

id	classCode	subClassCode	moodCode	code	codeVocId	codeOrig
3addd656-1b7a-0592-4615-9dc58be7eced	OBS		EVN	82711006	2.16.840.1.113883.6.96	408643008
ff2a06ee-5e4f-a647-62c7-19957ca1d311	SBADM		EVN	432102000	2.16.840.1.113883.6.96	432102000
type=TYPE;name=TRIAL_NAME;messageId=MESSAGE_ID2	FILE		EVN			

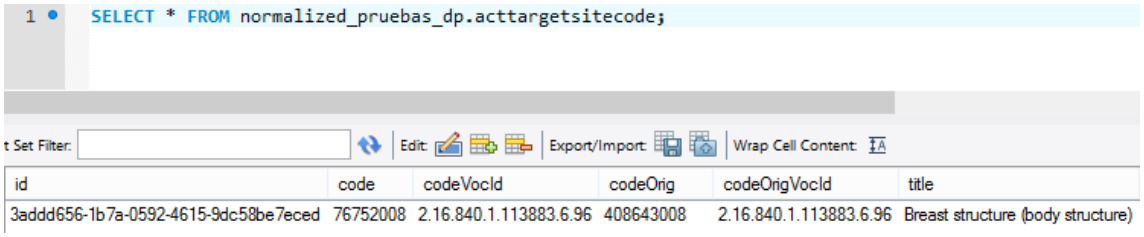
Figura 8.16: Tabla Act tras ejecución Data Push



```
SELECT * FROM normalized_pruebas_dp.actmethodcode;
```

id	code	codeVocId	codeOrig	codeOrigVocId	title
ff2a06ee-5e4f-a647-62c7-19957ca1d311	129445006	2.16.840.1.113883.6.96	432102000	2.16.840.1.113883.6.96	Administration - action (qualifier value)

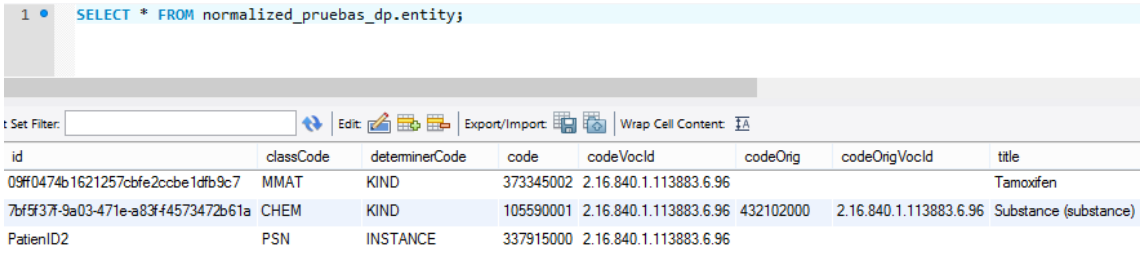
Figura 8.17: Tabla ActMethodCode tras ejecución Data Push



```
SELECT * FROM normalized_pruebas_dp.acttargetsitecode;
```

id	code	codeVocId	codeOrig	codeOrigVocId	title
3addd656-1b7a-0592-4615-9dc58be7eced	76752008	2.16.840.1.113883.6.96	408643008	2.16.840.1.113883.6.96	Breast structure (body structure)

Figura 8.18: Tabla ActTargetSiteCode tras ejecución Data Push



```
SELECT * FROM normalized_pruebas_dp.entity;
```

id	classCode	determinerCode	code	codeVocId	codeOrig	codeOrigVocId	title
09ff0474b1621257cbfe2ccbe1dfb9c7	MMAT	KIND	373345002	2.16.840.1.113883.6.96			Tamoxifen
7bf5f37f-9a03-471e-a83f-f4573472b61a	CHEM	KIND	105590001	2.16.840.1.113883.6.96	432102000	2.16.840.1.113883.6.96	Substance (substance)
PatienID2	PSN	INSTANCE	337915000	2.16.840.1.113883.6.96			

Figura 8.19: Tabla Entity tras ejecución Data Push

Como se puede observar en las figuras anteriores, no sólo se han creado las instancias de los actos presentes en el mensaje, sino que a su vez se han generado las instancias a almacenar en otras tablas del modelo común de datos representados por conceptos surgidos de la normalización.

9 CONCLUSIONES Y LÍNEAS FUTURAS

En esta última sección se recogen en primer lugar, las conclusiones obtenidas tras la finalización de este Trabajo de Fin de Grado, y en segundo lugar las posibles líneas futuras de desarrollo que permitan mejorar la herramienta.

9.1 Conclusiones

El objetivo principal de este Trabajo de Fin de Grado era proporcionar una herramienta que realice las funciones de normalización semántica de bases de datos que almacenan resultados de ensayos clínicos, obteniendo un conjunto de datos homogeneizado, permitiendo así, una consulta más precisa. Para ello, se han realizado unas fases de diseño, implementación y pruebas, cada una de ellas supeditada a la realización de la anterior.

La herramienta, proporcionada en forma de librería Java, recibe el nombre de la base de datos a normalizar y un fichero de configuración en su método de entrada, y realiza las siguientes tareas:

- **Preparar la base de datos normalizada.** La base de datos en la que se almacenarán los datos es creada si no existe con anterioridad, en el caso de que exista, se borrará su contenido o se mantendrá según convenga.
- **Copiar contenido invariable.** Se almacena el contenido que no sufre modificaciones en el proceso de normalización, asignando valores por defecto a campos que se encuentren vacíos en la base de datos original.
- **Normalización.** Se obtiene la forma normal corta de todos los códigos de conceptos en SNOMED-CT que representen actos o entidades y el enlace de los nuevos conceptos generados con las tablas de la base de datos. A su vez, se copia el contenido relacionado con dichos actos o entidades realizando comprobación de campos vacíos de la misma forma que se hacía en la tarea anterior.

El resultado final, es una base de datos esquematizada según el modelo común de datos de los proyectos INTEGRATE y EURECA, en la que el contenido se encuentra normalizado.

Cuando se dispone de bases de datos normalizadas, se permite desarrollar mecanismos de consulta más completos y precisos. El servicio Query Execution proporcionado en los proyectos en los que se integra el normalizador semántico, permite al usuario realizar consultas a las bases de datos de los proyectos. Las consultas realizadas son ejecutadas contras las bases de datos que han sido normalizadas con la herramienta desarrollada. Gracias a que el contenido consultado se encuentra representado en forma normal, se permite realizar consultas satisfactorias utilizando las plantillas SPARQL proporcionadas por otro servicio presente en INTEGRATE y EURECA, el Query Builder. El funcionamiento de este servicio es, dado un concepto SNOMED-CT,

construir una consulta haciendo uso de la forma normal de dicho concepto, es decir, no sólo añade en la consulta las líneas para consultar el concepto dado, generalmente se tratará de un tipo de acto, sino que a su vez, añade las líneas que consultan en alguna de las otras tablas de la base de datos los conceptos que forma parte de la representación en forma normal del concepto original. Si la base de datos en la que se ejecuta la consulta proporcionada por el Query Builder no se encontrase normalizada, dichas consultas no devolverían resultados.

Lo descrito anteriormente, nos da una de las razones principales por las que normalizar bases de datos, se podrá realizar un mayor número de consultas que devuelvan resultado. Tomemos como ejemplo el concepto en SNOMED-CT *234262008|Excision of axillary lymph*, se trata de un procedimiento médico que se almacena en la tabla *Act* del modelo común de datos. La representación en forma normal de este concepto produce dos nuevos, *129304002|Excision – action* y *68171009|Axillary lymph node structure*. El normalizador semántico se encargaría de almacenar estos dos nuevos conceptos en las tablas *ActMethodCode* y *ActTargetSiteCode* respectivamente, de esta forma, podríamos consultar el concepto original sólo, o con uno o ambos conceptos generados por la normalización obteniendo resultados, lo cual no ocurriría con una base de datos sin normalizar. De esta forma se obtendrán resultados más completos y con una mayor cantidad de información, además es posible consultar sólo por alguno de los nuevos conceptos producidos durante el proceso normalizador, así por ejemplo, podríamos consultar todos los pacientes en los que se les ha realizado algún método como *Axillary lymph node*.

La siguiente tabla recoge una comparativa del número de conceptos por los que se ha podido consultar información sobre pacientes y que han devuelto resultados entre una base de datos sin normalizar y la misma normalizada. Se han seleccionado algunos conceptos que pueden estar presentes en ambas bases de datos o sólo en la normalizada, en este último caso se trataría de conceptos generados por la normalización.

			#concepts retrieving data	
Concept	**Non-normalized CDM**	**Normalized CDM**	**Non-normalized CDM (expanded query)**	**Normalized CDM (expanded query)**
373377004|Moderately differentiated histological grade finding	1	1	17	17
172043006|Simple mastectomy	1	1	33	33
234262008|Excision of axillary lymph node	1	3	51	126
104934005|Sodium measurement, serum	1	3	28	102

Administration of substance 349848009|Epirubicin	1	3	16	56
118418003|Trocar device	0	1	0	17
129304002|Excision - action	0	1	0	18
68171009|Axillary lymph node structure	0	1	0	57
349848009|Epirubicin	0	1	0	16

La primera columna de la tabla recoge el concepto por el que se realizó la consulta. Los dos siguientes corresponden a realizar la consulta en la base de datos sin normalizar y normalizada respectivamente. Las dos últimas columnas recogen la misma información que las anteriores, pero esta vez se realiza la consulta utilizando un mecanismo de expansión de consultas proporcionada por el servicio Query Execution, que no sólo consulta por el concepto dado sino que consulta también por sus descendientes en la posición jerárquica que ocupe en la ontología SNOMED-CT, de esta forma se obtendrán resultados consultando tanto por el concepto original como por los padres de éste.

Los dos primeros conceptos consultados son conceptos primitivos, no se modifica su representación tras la normalización. Por tanto, se recupera información con el mismo número de conceptos consultados en ambas bases de datos.

Los tres siguientes conceptos consultados, *234262008*, *104934005* y *Administration of substance 349848009*, presentan una representación en forma normal en la que se añaden nuevos conceptos, por tanto no sólo se recuperará información preguntando por el concepto original y sus padres, sino también por los generados por el normalizador semántico y los padres de los mismos.

El resto de conceptos consultados son resultado de la normalización, por tanto, sólo se recuperará información consultando por dichos conceptos y sus padres en la base de datos normalizada.

En vista de estos resultados, podemos concluir que gracias a realizar consultas en bases de datos normalizadas no sólo obtendremos una información más completa, sino que a su vez podremos consultar dicha información de un mayor número de maneras posibles.

Teniendo en cuenta los resultados obtenidos en la etapa de pruebas del producto, y los objetivos planteados al comienzo de este documento, podemos afirmar que se han cumplido las metas previstas, ya que tras unas etapas iniciales de recopilación y análisis de información, se ha desarrollado con éxito la herramienta de normalización de bases de datos en ensayos clínicos y ha sido probada con éxito con datos reales. A su vez, se ha probado la integración de la misma en la capa de interoperabilidad semántica de los proyectos INTEGRATE y EURECA.

9.2 Líneas futuras de desarrollo

A pesar de que el correcto funcionamiento de la herramienta ya ha sido validada, es posible realizar una serie de mejoras en eficiencia, completitud o añadiendo funcionalidades.

Algunas de las posibles mejoras que se podrían incluir son las siguientes:

- Normalizar conceptos presentes en otras tablas del modelo. El normalizador semántico se encarga de normalizar conceptos almacenados en las tablas *Act* y *Entity* del modelo común de datos, sin embargo, existen otras tablas en las que se almacenan conceptos en SNOMED-CT en las cuales se puede estudiar si se obtendría beneficio al normalizar dichos conceptos.
- Crear hilos de ejecución encargados del almacenamiento de nuevas instancias en las tablas *ActMethodCode*, *ActTargetSiteCode* y *Entity*. Esta tarea se podría realizar de manera paralela a la ejecución principal del programa, mejorando la eficiencia y reduciendo los tiempos de ejecución de la herramienta.
- Añadir un mecanismo de traducción de conceptos entre terminologías. Como ya hemos dicho, los conceptos utilizados se encuentran codificados según las terminologías médicas SNOMED-CT, LOINC y HGNC, sin embargo, se podría incluir una tarea previa que traduzca conceptos presentes en la base de datos original a estas terminologías médicas en el caso de que no pertenezcan de antemano.

10 BIBLIOGRAFÍA

[1] Proyecto INTEGRATE. http://www.fp7-integrate.eu/index.php/project

[2] Proyecto EURECA. http://eurecaproject.eu/

[3] Paraiso S, Perez Del Rey D, Bucur A, Claerhout B, Alonso-Calvo R. Semantic normalization and query abstraction based on SNOMED-CT and HL7: Supporting multi-centric clinical trials. IEEE J Biomed Health Inform. 17 de Septiembre 2014

[4] Donnelly K. SNOMED-CT: The Advanced Terminology and Coding System for eHealth. Medical and care compunetics 3 2006: 279-290.

[5] College of American Pathologists. SNOMED Clinical Terms Guide. Transforming Expressions to Normal Forms. Agosto 2006. http://www.cap.org/apps/docs/snomed/documents/transformations_to_normal_forms.pdf

[6]McDonald CJ, Hu SM, Suico JG, Hill G, Leavelle D, Aller R, Forrey A, Mercer K, DeMoor G, Hook J, Williams W, Case J, Maloney P. LOINC, a Universal Standard for Identifying Laboratory Observations: A 5-Year Update. Clinical Chemistry. 2003; 49 No. 4: 624-633.

[7] HL7 International Organization. http://www.hl7.org/

[8] Wiki oficial de HL7. http://wiki.hl7.org/index.php?title=Main_Page

[9] HL7 RIM. http://www.hl7.org/implement/standards/rim.cfm

[10] Lenguaje Java. http://www.oracle.com/technetwork/java/index.html

[11] Conectores MySQL. http://www.mysql.com/products/connector/

[12] Framework Sesame. http://rdf4j.org/

[13] Gestor MySQL. http://www.oracle.com/us/products/mysql/overview/index.html

[14] RDF. http://www.w3.org/RDF/

[15] OWL. http://www.w3.org/2001/sw/wiki/OWL

[16] Lenguaje de consulta SPARQL. http://www.w3.org/TR/rdf-sparql-query/

[17] Especificación de Requisitos según el estándar de IEEE 830

https://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf

[18] Librería Log4j http://logging.apache.org/log4j/2.x/

[19] Página oficial de Apache Software Foundation http://www.apache.org/

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Wed Jan 07 19:25:34 CET 2015
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)